**UNIVERSIDADE FEDERAL DE SANTA CATARINA**
**PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS**

Victor Boeing Ribeiro

# EXPLORAÇÃO DE AMBIENTES DESCONHECIDOS ESTRUTURADOS COM SISTEMAS MULTI-ROBÔS

Florianópolis

2014

Victor Boeing Ribeiro

# EXPLORAÇÃO DE AMBIENTES DESCONHECIDOS ESTRUTURADOS COM SISTEMAS MULTI-ROBÔS

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas.
Orientador: Jean-Marie Farines, Dr.
Coorientador: José Eduardo Ribeiro Cury, Dr.

Florianópolis

2014

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Ribeiro, Victor Boeing
Exploração de ambientes desconhecidos estruturados com sistemas multi-robôs / Victor Boeing Ribeiro ; orientador, Jean-Marie Farines ; coorientador, José Eduardo Ribeiro Cury. - Florianópolis, SC, 2014.
120 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. Sistemas Multi-Robôs. 3. Mapeamento. 4. Planejamento de Trajetória. 5. Alocação de Tarefas. I. Farines, Jean-Marie. II. Cury, José Eduardo Ribeiro. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Victor Boeing Ribeiro

**EXPLORAÇÃO DE AMBIENTES DESCONHECIDOS ESTRUTURADOS COM SISTEMAS MULTI-ROBÔS**

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre em Engenharia de Automação e Sistemas", e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Florianópolis, 28 de fevereiro 2014.

---

Jomi Fred Hübner, Dr.
Coordenador

---

Jean-Marie Farines, Dr.
Orientador

---

José Eduardo Ribeiro Cury, Dr.
Coorientador

**Banca Examinadora:**

---

Jean-Marie Farines, Dr.
Presidente

---

Ubirajara Franco Moreno, Dr.

Rodrigo Tacla Saad, Dr.

Max Hering de Queiroz, Dr.

*Ao infinito... e além!*

## RESUMO

A busca e exploração em ambientes desconhecidos é um problema fundamental da robótica móvel. Para a execução da exploração de um ambiente, desde o ato de mover-se até a construção de um mapa, existem diversos aspectos que, separadamente, constituem importantes temas de pesquisa. A utilização de sistemas multi-robôs pode fornecer diversas vantagens para a resolução desse tipo de problema, porém, adicionam novos desafios, como a coordenação dos múltiplos robôs e a distribuição de tarefas entre eles. Este trabalho apresenta um método de exploração de ambientes desconhecidos estruturados com múltiplos robôs, tratando das etapas de mapeamento e planejamento de trajetórias. Enquanto exploram o ambiente, os robôs interpretam informações sensoriais inserindo-as em um mapa topológico global único. Dados métricos são também coletados e incorporados ao mapa, facilitando a resolução de problemas como o de fechamento de *loops*. O método proposto utiliza o conceito de *fronteiras* para definir as tarefas do sistema e avalia a utilização de heurísticas na alocação das tarefas aos robôs. Uma abordagem desacoplada é utilizada para o planejamento de trajetória, na qual caminhos são calculados para cada robô e conflitos entre os mesmos são tratados em uma etapa posterior. Resultados experimentais validam a funcionalidade e eficiência do método proposto.


**Palavras-chave:** sistemas multi-robôs, mapeamento, planejamento de trajetória, alocação de tarefas

## ABSTRACT

The search and exploration in unknown environments is a fundamental problem in mobile robotics. For the execution of the environment exploration, since the act of moving to the construction of a map, there are several aspects, which individually are important research topics. The use of multi-robot systems can provide several advantages for solving this type of problem, however, adds new challenges such as the coordination of multiple robots and the task distributon between them. This work presents a method of exploring unknown structured environments with multiple robots, dealing with the path planning and mapping stages. While exploring the environment, the robots interpret sensorial information inserting them into a single global topological map. Metric data are also collected and incorporated into the map, making easier the resolution of problems as the loop closing. The proposed method uses the concept of *frontiers* to define the tasks of the system and evaluates the use of heuristics to allocate tasks to the robots. A decoupled approach is used for path planning, in which paths are calculated for each robot and then conflicts between them are resolved. Experimental results validate the functionality and efficiency of the proposed method.


**Keywords:** multi-robot systems, mapping, path planning, task allocation

## LISTA DE FIGURAS

## SUMÁRIO

# 1 INTRODUÇÃO

A busca e exploração em ambientes desconhecidos é um problema fundamental da robótica móvel. Existem diversas aplicações para esse tipo de problema, como o reconhecimento de terrenos [Hougen et al. 2000], operações de resgate [Thrun et al. 2003], limpeza [Lee, Baek e Oh 2011] e vigilância [Vallejo et al. 2009]. O objetivo principal, nesses casos, é a exploração de um ambiente do qual se tem informações limitadas, buscando a localização de pontos específicos ou sua exploração completa e, em geral, exigindo a construção de um mapa que possibilite a execução de outras atividades.

Para a execução da exploração de um ambiente, desde o ato de mover-se até a construção de um mapa, existem diversos aspectos que, separadamente, constituem importantes temas de pesquisa. Esses aspectos abrangem desde a percepção do ambiente até tomadas de decisão em relação a qual caminho seguir, impactando diretamente no sucesso e eficiência dessa exploração.

Diversas soluções têm sido propostas para a resolução desse tipo de problema. Na medida em que essas soluções apresentam resultados cada vez melhores para sistemas com um único robô, especialmente para ambientes livres de obstáculos, o próximo passo é estender sua resolução para grupos de robôs móveis em ambientes estruturados [Marjovi e Marques 2011].

A utilização de sistemas multi-robôs pode fornecer diversas vantagens sobre sistemas com um único robô, como o aumento da velocidade de exploração, melhor precisão alcançada com redundância de sensores, tolerância à falta, etc. Contudo, em adição aos problemas existentes em sistemas com um único robô, a extensão a sistemas multi-robôs coloca alguns novos desafios. Entre esses desafios destacam-se a comunicação entre robôs e com uma estação central, a determinação de uma arquitetura que deve tratar das interações entre robôs, a distribuição das tarefas que compõe o objetivo global do sistema e a coordenação da movimentação dos múltiplos robôs evitando colisões e bloqueios.

Entre os desafios apresentados, dois recebem atenção especial nesse trabalho: a distribuição das tarefas e a coordenação da movimentação dos múltiplos robôs. O primeiro, denominado de *alocação de tarefas*, refere-se à determinação de qual robô deve visitar cada ponto do mapa. O segundo, denominado *coordenação de sistemas multi-robôs*, refere-se à geração de trajetórias para os robôs coordenando sua movimentação para prevenir colisões e bloqueios. Tanto a alocação de ta-

refas quanto a coordenação de sistemas multi-robôs estão diretamente ligadas ao sucesso e eficiência do sistema. Assim, como é desejável que a exploração do ambiente desconhecido ocorra dentro do menor período de tempo possível, evitar a realização de trabalho redundante é um aspecto fundamental na eficiência do sistema.

A alocação de tarefas deve levar em consideração, por exemplo, o custo para atingir determinado ponto do mapa, como a distância a ser percorrida e o gasto de energia necessário. A utilidade também é um aspecto importante, representando a quantidade de informação que a visita a um ponto irá fornecer. A existência de múltiplos robôs explorando o ambiente pode ser trabalhada na alocação de tarefas nessa mesma função custo/utilidade: quando existem outros robôs em determinada região do ambiente, o custo para atingir esse ponto pode ser elevado e a utilidade diminuída, buscando distribuir os robôs pelo ambiente. Em [Burgard et al. 2005], por exemplo, a utilidade de uma região do ambiente é inversamente proporcional ao número de robôs presentes nessa região. Já em [Gerkey e Matarić 2004] a utilidade da realização de uma tarefa é medida pela diferença entre sua qualidade, como a confiabilidade do mapa resultante da exploração por determinado robô, e o custo.

A coordenação de sistemas multi-robôs ocorre juntamente com o cálculo de trajetória. Ela deve atender especificações de segurança, evitando bloqueios e colisões, e também a aspectos de eficiência do sistema. Trajetos com desvios desnecessários são, então, indesejados. Diversas técnicas propõem soluções para esse problema, como em [Quottrup, Bak e Zamanabadi 2004], onde técnicas de verificação formal são utilizadas para obtenção de trajetórias coordenadas. Um aspecto a ser considerado é que, na resolução do problema de exploração, o cálculo de trajetórias coordenadas ocorre diversas vezes, já que cada robô deve explorar vários pontos do ambiente. A velocidade desse cálculo é, então, um dos critérios que impacta na eficiência do sistema.

## 1.1 OBJETIVOS

O objetivo principal dessa dissertação consiste em propor um método de exploração de ambientes desconhecidos estruturados com múltiplos robôs. O método proposto deve tratar dos dois aspectos principais que compõe o problema de exploração: o mapeamento e o planejamento de trajetória.

O método proposto para o problema de exploração deve abordar

também os desafios gerados pela resolução desse problema com sistemas multi-robôs. Entre esses desafios se destacam o de *alocação de tarefas* e o de *coordenação de sistemas multi-robôs*, por desempenharem papel fundamental na eficiência do sistema.

No caso da alocação de tarefas, o objetivo é propor heurísticas que diminuam as interações entre robôs e a passagem destes por regiões já exploradas do ambiente. A alocação de tarefas é realizada buscando a tarefa de menor custo para determinado robô. Dessa forma, as heurísticas são inseridas no comportamento do sistema modificando os custos de atingir determinadas regiões do ambiente.

No caso da coordenação, que é resolvida dentro do planejamento de trajetórias, o objetivo é propor soluções para que trajetórias sejam geradas respeitando especificações de segurança. Essas soluções devem oferecer uma alternativa ao problema de escalabilidade presente em métodos centralizados.

Um estudo de caso deverá ser realizado a partir do método proposto, no qual os múltiplos robôs devem explorar um ambiente com características específicas. Este estudo deve abranger todas as etapas do problema de exploração, para as quais serão propostas soluções.

## 1.2 RESULTADOS ESPERADOS

Os resultados esperados deste trabalho se concentram principalmente na apresentação de um método de exploração de ambientes desconhecidos em particular. Este método deve tratar de soluções para os dois aspectos principais do problema de exploração: o mapeamento e o planejamento de trajetória. Deve tratar também dos desafios gerados pela resolução do problema de exploração com múltiplos robôs, onde se destacam a alocação de tarefas e a coordenação de sistemas multi-robôs. Os resultados esperados estendem-se também às etapas de implementação e validação, realizadas através de simulação do sistema e análise do seu desempenho através do tempo total de exploração. Os principais resultados esperados são listados abaixo.

1. Um dos resultados esperados é a proposição de um método de exploração de ambientes desconhecidos com sistemas multi-robôs. Soluções para todas as etapas do problema de exploração serão apresentadas nesse método;

2. Proposição de uma abordagem para realização do mapeamento de um ambiente estruturado com características específicas. A

abordagem em questão deve ser adequada a robôs de construção simples, não devendo depender de altas taxas de processamento e unidades sensoriais sofisticadas para extração das características do ambiente;

3. Emprego de diferentes heurísticas para a alocação de tarefas que reduzam a passagem de robôs por áreas já exploradas do ambiente e o número de interações entre robôs, com intuito de reduzir o tempo total de exploração. Espera-se que o emprego do alocador desenvolvido apresente maior eficiência que uma alocação realizada de forma aleatória;

4. Desenvolvimento de um algoritmo de planejamento de trajetória que seja capaz de coordenar os robôs de forma segura. Abordagens centralizadas combinam o espaço de configuração de todos os robôs em um único espaço de configuração, no qual é realizada uma busca para solucionar o problema como um todo. Essas abordagens fornecem soluções ótimas para o problema de coordenação, porém, apresentam aumento exponencial do espaço de buscas com o aumento do sistema (número de robôs, por exemplo). É esperado que o planejador proposto apresente uma melhor relação entre tempo de processamento e custo do caminho gerado (comprimento), através de uma abordagem desacoplada;

5. Implementação das técnicas de exploração propostas em um ambiente de simulação que possibilite comparar e validar as mesmas, mensurando a eficiência do sistema atuando em diferentes situações, como ambientes, número de robôs, etc. Para tal, é proposto um estudo de caso no qual múltiplos robôs executam a atividade de exploração em um ambiente estruturado com características específicas. Todas as etapas da atividade de exploração devem estar presentes nesse estudo de caso.

## 1.3 ESTRUTURA DA DISSERTAÇÃO

Os próximos capítulos estão organizados da seguinte forma:

**Capítulo 2** apresenta os diversos desafios no contexto da exploração de ambientes desconhecidos por robôs móveis autônomos. As principais vantagens da utilização de sistemas multi-robôs para realização desse tipo de atividade são também apresentadas, assim como os principais desafios que surgem da união desses dois

temas: o problema de exploração com múltiplos robôs. O capítulo ainda revisa os principais trabalhos relacionados ao tema e descreve as características do ambiente que será utilizado ao longo deste trabalho para exemplificar as técnicas desenvolvidas;

**Capítulo 3** apresenta as principais etapas do problema de construção de mapas e os desafios gerados pela sua combinação. O capítulo revisa as principais abordagens de mapeamento e de planejamento de trajetória e apresentada técnicas de discretização do mapa obtido na etapa de mapeamento para a posteriormente planejar a trajetória dos robôs;

**Capítulo 4** trata do método de exploração de ambientes desconhecidos proposto nesse trabalho, apresentando as abordagens selecionadas para o mapeamento e o planejamento de trajetória. São apresentados o algoritmo para coordenação dos múltiplos robôs, no contexto do planejamento de trajetória, e heurísticas utilizadas na alocação de tarefas para aumento da velocidade de exploração.

**Capítulo 5** trata de aspectos da implementação do sistema em ambiente de simulação e realiza diversos experimentos variando ambiente, número de robôs e técnica empregada. Os resultados dos experimentos são, então, analisados e comparados;

**Capítulo 6** apresenta as conclusões e perspectivas do trabalho desenvolvido.

# 2 A EXPLORAÇÃO DE AMBIENTES DESCONHECIDOS

A atividade de busca e exploração em ambientes desconhecidos é um dos problemas fundamentais em sistemas robóticos autônomos. São diversas as aplicações desse tipo de sistema, desde a substituição de operadores humanos em atividades cuja execução apresenta elevado risco, até a automação de atividades cotidianas. Em [Nagatani et al. 2011], por exemplo, a realização do mapeamento de ambientes por robôs é utilizada para auxiliar equipes de resgate em situações de desastres. Já em [Lee, Baek e Oh 2011], o problema de exploração é tratado dentro de um contexto de robôs autônomos realizando limpeza em ambientes domésticos. Devido às diversas possibilidades de aplicações, a capacidade de se mover em um ambiente desconhecido e construir um modelo representativo do ambiente tem sido intensivamente estudada. Os principais desafios nessa área são a manutenção da localização do robô durante a exploração, a adequação de modelos de representação do ambiente e a determinação de estratégias para a movimentação do robô.

O uso de múltiplos robôs geralmente apresenta diversas vantagens sobre sistemas com um único robô [Burgard et al. 2005]. Primeiramente, sistemas multi-robôs têm o potencial de realizar tarefas mais rapidamente. Além disso, eles possuem maior tolerância a faltas que um único robô. Outra vantagem é a redundância de informações que ajuda a lidar com incertezas. Quando a exploração de ambientes desconhecidos é realizada simultaneamente por múltiplos robôs, essas características são levadas em conta para o comportamento do sistema. No entanto, desafios inerentes a sistemas multi-robôs passam também a compor o problema de exploração como, por exemplo, a necessidade de coordenar a movimentação dos múltiplos robôs para evitar colisões e bloqueios.

Nesse capítulo, inicialmente são apresentados os principais aspectos referentes à exploração de ambientes desconhecidos por sistemas multi-robôs coordenados. Em seguida, é apresentada uma revisão de trabalhos e técnicas inseridos nesse contexto. Por fim, são descritas as características do ambiente desconhecido adotado para aplicação das técnicas propostas no decorrer desse trabalho.

## 2.1 INTRODUÇÃO AO PROBLEMA DE EXPLORAÇÃO

De acordo com [Yamauchi 1997], a exploração de um ambiente desconhecido é definida como:

> *"O ato de mover-se através de um ambiente desconhecido enquanto realiza a construção de um mapa para subsequente navegação. Uma boa estratégia de exploração é aquela que gera um mapa completo, ou quase completo, em uma quantidade razoável de tempo."*

Para a execução da exploração de um ambiente, desde o ato de mover-se até a construção de um mapa, existem diversos aspectos que, separadamente, constituem importantes temas de pesquisa. Esses aspectos abrangem desde a percepção do ambiente até tomadas de decisão em relação a qual caminho seguir, impactando diretamente no sucesso e eficiência dessa exploração.

Em [Thrun et al. 2002] são apontados os principais aspectos considerados como desafios na tarefa de exploração e construção do mapa de um ambiente desconhecido. O primeiro deles é a *aquisição de informações* dos arredores do robô, com sensores capazes de mensurar grandezas para a montagem e manutenção de um modelo que represente o ambiente. Entre os sensores comumente utilizados para essa tarefa estão câmeras, sensores de distância baseados em ultrassom e infravermelho, laser, sensores tácteis, bússolas, acelerômetros, *encoders*, etc. Todos esses sensores apresentam limitações em relação ao alcance: apenas uma pequena porção ao redor do robô é de fato medida, fazendo-se necessário que o robô explore o restante do ambiente para obter uma informação global. Além disso, sensores estão sujeitos a ruídos e erros de medição, de distribuição frequentemente desconhecida, que podem levar a conclusões errôneas a respeito do ambiente.

Vale destacar que os problemas existentes na aquisição de informações podem influenciar na localização do robô, que é um aspecto fundamental na exploração de ambientes desconhecidos. A localização está sujeita aos mesmos erros e ruídos de medição, porém, acrescida à inexatidão da movimentação dos robôs. Efeitos como derrapagens e deslizamentos (*drift/slippage*) geram erros na odometria, e um pequeno erro de medição ao se executar um movimento rotacional, por exemplo, pode gerar grandes erros na estimação da posição do robô.

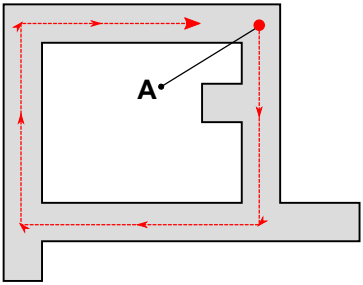
Um segundo desafio refere-se à *alta complexidade do ambiente* sendo mapeado. A grande quantidade de formas que podem estar presentes em um ambiente é um fator que torna a atividade de exploração

ainda mais complexa. Esse problema pode ser observado ao se considerar o grande número de objetos e estruturas existentes em um ambiente, cuja descrição é necessária para a criação de um mapa detalhado. Em geral, para lidar com esse problema, apenas algumas formas principais são extraídas do ambiente e descritas no mapa, tais como corredores, salas ou esquinas.

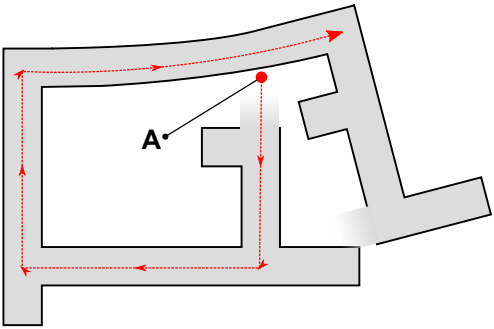
Outro desafio a ser considerado refere-se ao problema de *determinação de equivalência* de um ponto quando este é mensurado em períodos de tempo diferente. Como a medição da localização é inexata, existe uma grande dificuldade na determinação de quando o robô fechou um "loop" em ambientes que apresentam estruturas cíclicas, ou seja, quando ele voltou a um ponto já visitado, porém, através um caminho diferente. Erros incrementais na localização do robô podem gerar incertezas de magnitude muito elevada com o passar do tempo, fazendo com que o problema de determinação de equivalência entre dois pontos tenha alta complexidade. Diversos trabalhos tratam especificamente desse assunto, como [Beevers e Huang 2005], [Ji et al. 2009] e [Savelli e Kuipers 2004]. Na Figura 1 é mostrado um exemplo do resultado de um erro de determinação de equivalência de um ponto. Partindo do ponto A, o robô percorre o trajeto indicado na figura retornando ao mesmo ponto. A equivalência do ponto A é interpretada corretamente na Figura 1(a) e erroneamente como um segundo ponto na Figura 1(b).

Outro desafio está ligado à *dinamicidade do ambiente*. Mudanças nos arredores do robô durante a execução de uma exploração geram inconsistências nas leituras dos sensores, fazendo com que essas possam ser mal interpretadas, podendo levar o sistema a crer que está num lugar diferente do que realmente está. A possibilidade do fechamento de uma porta, passagem de pessoas ou movimentação de um móvel criam um desafio bastante grande. Em geral, trabalhos na área de exploração de ambientes desconhecidos desconsideram a dinamicidade do ambiente ou a tratam de forma isolada e com comportamento conhecido. Essa dissertação considera apenas um ambiente estático, não tratando do problema de dinamicidade.

Um último desafio refere-se à determinação de uma *estratégia de movimentação* para a escolha de qual caminho seguir durante a exploração. Quando um modelo do ambiente é conhecido, técnicas completas e ótimas podem ser aplicadas para planejar o deslocamento do robô de um ponto a outro da forma mais eficiente, porém, sem conhecimento do ambiente ou com um conhecimento parcial, essas técnicas não podem ser utilizadas. Técnicas de planejamento de exploração, em geral, tomam decisões através de avaliações de fatores como os ganhos de




(a) Trajeto percorrido e interpretação correta de correspondência de um ponto




(b) Interpretação errônea da correspondência de um ponto, causando distorção no mapa gerado

Figura 1: Exemplo de erro no fechamento de um *loop*

determinado ponto para a construção do mapa, ou a energia que será gasta para se deslocar até o mesmo. Dessa forma, as soluções existentes para o planejamento da exploração apresentam resultados sub-ótimos.

Uma visão geral dos principais desafios existentes na realização da atividade de exploração de ambientes desconhecidos é apresentada abaixo.

- Aquisição de informações: sensores utilizados para mensurar grandezas do ambiente possuem alcance limitado e apresentam ruídos e erros de medição;
- Alta complexidade do ambiente: ambientes são comumente formados por uma grande quantidade de formas, como objetos e estruturas, o que torna complexa a atividade de geração de um

mapa completo;

- Determinação de equivalência de pontos: dificuldade de determinar a equivalência de um ponto do ambiente quando este é alcançado através de caminhos diferentes;
- Dinamicidade do ambiente: ambientes, em geral, se modificam com o passar do tempo. O fechamento de uma porta ou a passagem de pessoas, por exemplo, pode gerar inconsistências nas leituras dos sensores;
- Estratégia de movimentação: determinação de qual caminho seguir para executar a exploração.

Dentre os desafios apresentados, o que recebe atenção especial no método proposto neste trabalho é o de determinação de uma estratégia de movimentação. São também propostas soluções para os desafios de aquisição de informações e determinação de equivalência. Entretanto, o ambiente estudado é considerado estático, eliminando a necessidade de trabalhar com aspectos de dinamicidade do ambiente. É também assumindo um ambiente estruturado sem obstáculos, diminuindo a sua complexidade.

## 2.2 SISTEMAS MULTI-ROBÔS NA EXPLORAÇÃO

Um robô móvel autônomo é um sistema fisicamente independente, equipado com sensores e atuadores suficientes para realizar uma dada tarefa. Os sistemas multi-robôs são, então, compostos por robôs móveis autônomos que operam conjuntamente em um ambiente, realizando uma ou mais tarefas específicas.

Diversas são as características de sistemas multi-robôs que justificam sua utilização no lugar de sistemas com um único robô e, consequentemente, motivam o desenvolvimento e estudo de novas soluções. De acordo com [Siciliano e Khatib 2008], algumas das principais motivações para o desenvolvimento desse tipo de sistema são:

- A complexidade da tarefa é muito elevada para ser realizada por um único robô;
- A tarefa é inerentemente distribuída;
- A facilidade da construção de vários robôs com recursos limitados em relação à construção de um único robô dotado de todas as habilidades necessárias;

- Múltiplos robôs podem solucionar problemas mais rapidamente realizando tarefas paralelas;

- Múltiplos robôs fornecem um aumento da robustez de sistemas através de redundância de informações, resultando também em uma tolerância a faltas.

A partir das vantagens apresentadas para esse tipo de sistema, três se destacam no caso de exploração de ambientes desconhecidos: o potencial de executar tarefas mais rapidamente, a redundância de informações que ajuda a lidar com incertezas e a maior tolerância a faltas. A primeira delas, o aumento da velocidade de exploração, é justificada pelo fato de existir a possibilidade de explorar paralelamente diferentes regiões do ambiente. A segunda refere-se a um ganho na exatidão das medições realizadas através de redundância, ou seja, a medição de uma grandeza realizada por mais de um robô ajuda a compensar os efeitos de incerteza dos sensores, especialmente quando os robôs estão equipados com sensores diferentes. A terceira vantagem é a tolerância a falta: mesmo com a falta de um dos robôs o sistema pode continuar a realização da exploração.

Os sistemas multi-robôs, apesar de serem potencialmente muito eficientes, possuem uma série de questões a serem consideradas para um real aproveitamento de suas funções. Entre essas questões se destacam:

- Arquitetura;

- Comunicação;

- Alocação de tarefas;

- Coordenação.

O modelo de arquitetura utilizado para controlar múltiplos robôs tem um impacto significativo na robustez e escalabilidade do sistema. Esse modelo de arquitetura é composto pelos mesmos componentes presentes na arquitetura de sistemas com um único robô, porém, deve também tratar das interações entre os robôs e da determinação do comportamento global do sistema [Siciliano e Khatib 2008].

A comunicação é outro aspecto fundamental na concepção de sistemas multi-robôs. Para alcançar soluções coerentes e eficientes é desejável o acesso à maior quantidade possível de informações do sistema. Essas informações, no entanto, estão espalhadas entre os robôs, existindo, então, a necessidade de comunicação. Como exemplo, temos

a tarefa de exploração. Quanto mais informações a respeito da localização dos robôs, das tarefas que eles estão executando e dos mapas locais construídos por cada um deles, mais eficiente será o planejamento das próximas ações.

Em geral, aplicações envolvendo sistemas multi-robôs são caracterizadas por um conjunto de tarefas que juntas compõe o objetivo global da aplicação. Uma vez definidas essas tarefas, o desafio concentra-se em determinar qual robô deve executar cada uma delas; este desafio é chamado de *alocação de tarefas*. O problema da alocação de tarefas possui diversas variações de acordo com a aplicação em questão. O número de robôs necessários para realizar uma tarefa e a dependência entre as mesmas são fatores que influenciam na alocação. Quando cada um dos robôs possui diferentes capacidades, por exemplo, existem tarefas que só podem ser executadas por um tipo específico de robô.

Quando múltiplos robôs estão realizando tarefas em um ambiente compartilhado, existe a necessidade de coordenar suas ações para prevenir colisões e bloqueios. Essa necessidade se torna ainda mais evidente quando o ambiente em questão possui regiões de gargalo e poucas áreas abertas, permitindo muitas vezes a passagem de apenas um robô. Ambientes estruturados, como os utilizados no decorrer desse trabalho, ilustram essa necessidade.

Neste trabalho, os aspectos que efetivamente serão levados em conta na proposição do método de exploração se referem à alocação de tarefas e à coordenação dos múltiplos robôs. Para a alocação de tarefas é proposto um método que incorpora heurísticas com intuito de minimizar a passagem dos robôs por regiões já exploradas do ambiente e a interferência entre eles, diminuindo o tempo total necessário para a exploração completa do ambiente. Já para a coordenação dos múltiplos robôs é proposto um método de cálculo de caminhos até as tarefas, garantindo propriedades de segurança e buscando também a diminuição do tempo de exploração.

## 2.3 TRABALHOS RELACIONADOS

Muitos trabalhos têm sido desenvolvidos no contexto de exploração e mapeamento de ambientes desconhecidos. Porém, devido aos diversos desafios inerentes a esse tema, esses trabalhos geralmente se focam em um dos problemas ou até mesmo em aplicações específicas.

Em [Marjovi e Marques 2011] são aplicadas técnicas de exploração de ambientes desconhecidos com múltiplos robôs para identificação

de fontes de vazamento de gás em ambientes industriais. Sua principal contribuição é a utilização da medição de concentração de odores e de direção de deslocamento de ar para decidir os caminhos a serem tomados pelos robôs, buscando aqueles com maiores chances de levarem a uma fonte de vazamento. [Burgard et al. 2005] segue na mesma direção, trabalhando na determinação dos pontos de maior utilidade do ambiente durante a atribuição de atividades para os múltiplos robôs. Nesse trabalho, a ideia principal consiste em evitar que vários robôs se movam para uma mesma região, fazendo com que a utilidade de um ponto seja inversamente proporcional à quantidade de robôs presente nos seus arredores. Também é abordado o fato de que os robôs possuem comunicação limitada, não estando aptos a se comunicar em qualquer instante de tempo.

Muitos dos trabalhos desenvolvidos na área trabalham com o conceito de "fronteiras", proposto num contexto de exploração com um único robô em [Yamauchi 1997] e num contexto de sistemas multi robôs em [Yamauchi 1998]. Nessa abordagem, os robôs são enviados para os limites entre regiões exploradas e ainda não exploradas do ambiente, sendo esses pontos "alvos" a serem visitados pelos robôs. Como uma complementação ao método de fronteiras existe o método de segmentação do ambiente, usado em [Wurm, Stachniss e Burgard 2008], por exemplo. A ideia é segmentar a área já explorada, enviando os robôs para os segmentos ao invés de diretamente às fronteiras, objetivando distribuí-los melhor ao longo do ambiente, evitando interferência entre robôs e trabalho redundante. Em [Stachniss, Mozos e Burgard 2006] a atribuição de um alvo a ser explorado considera também o tipo de lugar em que este está inserido antes de designar um robô para explorá-lo, como corredores ou salas. A prioridade é visitar lugares que potencialmente forneçam mais lugares alvo a serem explorados, possibilitando enviar os robôs para regiões distantes umas das outras.

O problema do planejamento de trajetória dentro de um contexto de exploração multi-robôs geralmente é resolvido dentro de funções custo que, ao avaliarem os pontos a serem explorados, adicionam custos à localização de outros robôs, resultando na escolha de outro ponto ou no cálculo de um caminho com desvio, como em [Burgard et al. 2005]. Outras vezes, o planejamento de trajetória é feito desconsiderando a existência de outros robôs e problemas relacionados a possíveis colisões são tratados de forma reativa. Em [Marjovi e Marques 2011], quando dois robôs se encontram em direções contrárias durante a exploração, um deles permanece parado enquanto o outro desvia, baseando-se numa ordem de prioridades estabelecida.

Técnicas mais complexas de planejamento de trajetória têm sido desenvolvidas num contexto onde múltiplos robôs, a partir de um ponto inicial, devem atingir pontos especificados no ambiente. Em [Pavei 2011] são apresentadas técnicas de planejamento de trajetória baseadas da teoria de controle supervisório e também utilizando autômato-jogo temporizado. Em [Carvalho 2012], o problema de planejamento de trajetória é tratado através de métodos baseados em estruturas de jogos. Ambos os trabalhos consideram a existência de obstáculos dinâmicos em lugares conhecidos do ambiente, gerando coordenadores capazes de lidar com, além dos obstáculos, especificações que garantam segurança do sistema.

A verificação formal de modelos (*model checking*) é utilizada em [Quottrup, Bak e Zamanabadi 2004] para a geração de contra-exemplo usado para a coordenação dos robôs. Além disso, propriedades de segurança e vivacidade são garantidas através da verificação das mesmas.

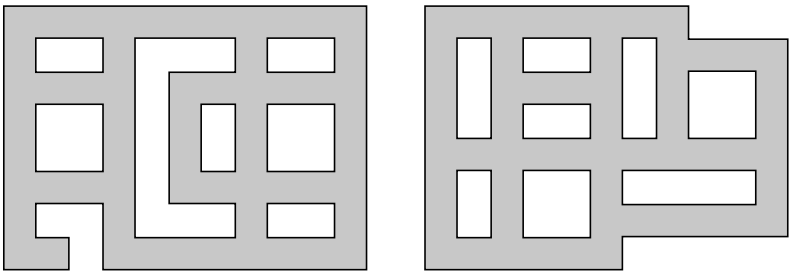
Em [Bennewitz 2004] é apresentado um método "desacoplado" de cálculo de trajetória para múltiplos robôs, apresentando uma alternativa ao problema de escalabilidade de métodos centralizados. Neste trabalho, é definida uma ordem de prioridades entre os robôs e um planejamento de trajetória é executado para cada um deles. O cálculo da trajetória de um robô considera apenas a trajetória dos robôs de mais alta prioridade, incluindo desvios quando existe interferência entre eles.

## 2.4 CARACTERIZAÇÃO DO PROBLEMA DE EXPLORAÇÃO E DO AMBIENTE DE TESTE

O problema de exploração de ambientes desconhecidos é abordado nesse trabalho tratando desde a aquisição e interpretação dos dados sensoriais coletados pelos robôs até a construção de um mapa representativo do ambiente. O foco, entretanto, é a alocação de tarefas e planejamento de trajetórias. Para isso, é considerado um ambiente simplificado e estático, uma arquitetura de coordenação centralizada e comunicação irrestrita.

Para exemplificar as técnicas abordadas nessa dissertação é assumido um ambiente com características específicas. A área a ser explorada pelos robôs é um ambiente *indoor* e estruturado, como encontrado em armazéns e escritórios. Como comumente encontrado em ambientes *indoor*, nenhum sistema de posicionamento global está disponível. A única informação existente sobre o ambiente é que se trata de uma

construção estruturada (que pode ser descrita através de primitivas geométricas), contendo corredores, esquinas, junções e cruzamentos. É assumido que, como encontrado usualmente em construções estruturadas, o ambiente é formado somente por paredes paralelas e ortogonais. Neste caso, a largura de um corredor é constante ao longo de sua extensão, assim como esquinas apresentam sempre mudanças de 90 graus na direção. Na Figura 2 são mostrados dois exemplos de ambiente com as características citadas.



(a) (b)

Figura 2: Exemplos de ambientes estruturados

O ambiente foi assumido ser dessa forma, pois, mesmo com a utilização de sensores relativamente simples, é possível identificar as estruturas do ambiente de maneira também simples. Outros trabalhos, como [Marjovi e Marques 2011] fazem a mesma consideração a respeito do ambiente. A utilização de sensores mais sofisticados como câmeras e lasers, conjuntamente com técnicas de reconhecimento compatíveis com os mesmos, possibilitaria, no contexto deste trabalho, a exploração de ambientes mais complexos.

A consideração de que o ambiente é formado somente por paredes paralelas e perpendiculares é, porém, uma consideração bastante realista quando se trata de um ambiente *indoor* estruturado. Em [Bando e Yuta 2010] essas características são não só assumidas como frequentemente encontradas, como também são utilizadas para correção de erros na estimação da direção dos robôs. O próprio ser humano utiliza-se dessas informações para estimação de sua localização quando navegando em um ambiente estruturado desconhecido.

Dentro desse ambiente, considera-se um grupo de N robôs móveis, equipados com sonares e *encoders*, movendo-se em um plano. Esses robôs são capazes de se comunicar entre eles e também com uma estação central responsável pela coordenação dos mesmos. Os robôs

são também capazes de identificar a presença de outros robôs quando aproximam-se frontalmente.

É assumido que os robôs possuem, no início da exploração, uma referência global, fazendo com que seus sistemas de coordenadas sejam os mesmos e que as informações coletadas do ambiente sejam diretamente incorporadas a um único mapa global. Essa referência comum pode ser a informação da posição relativa entre os pontos de partida dos robôs; os pontos de partida estarem localizados em uma região já conhecida do mapa ou assumir que os robôs partem do mesmo ponto do mapa em momentos diferentes.

Alguns trabalhos tratam do problema de exploração sem assumir uma referência global comum para os robôs [Marjovi e Marques 2011]. Nesses casos, são construídos mapa locais para cada um deles e utilizadas técnicas para fusão dos mesmos. Essa fusão pode ocorrer por similaridade dos mapas locais, busca de um referencial conhecido no ambiente ou até mesmo o encontro intencional dos robôs para alinhamento dos seus sistemas de coordenadas. O problema de fusão de mapas locais, não tratado neste trabalho, é complementar ao de planejamento de trajetória.

Dentro das condições estabelecidas em relação aos robôs e ao ambiente, tem-se como objetivo principal a exploração de todo o ambiente com os múltiplos robôs e a geração de um mapa global no menor tempo possível.

## 2.5 CONCLUSÃO

Neste capítulo foram apresentados os principais desafios existentes na realização da atividade de exploração de ambientes desconhecidos e as principais características que sistemas multi-robôs adicionam a essa atividade.

Os cinco principais desafios na exploração de ambientes desconhecidos são apontados em [Thrun et al. 2002]. São eles: a aquisição de informações, a alta complexidade do ambiente, a determinação de equivalência entre regiões, a dinamicidade e a determinação de uma estratégia de movimentação.

Entre as vantagens da utilização de sistemas multi-robôs na exploração se destacam: o potencial de executar atividades mais rapidamente devido ao paralelismo, a redundância de informações que ajuda a lidar com incertezas e a maior tolerância a faltas. Sistemas multi-robôs, entretanto, também adicionam novos desafios ao problema de

exploração. Os principais desafios a serem considerados são relacionados à arquitetura do sistema, comunicação, alocação de tarefas e coordenação dos múltiplos robôs.

Na sequência, foram apresentados trabalhos inseridos no contexto de exploração de ambientes desconhecidos e de coordenação de sistemas multi-robôs. Técnicas mais complexas de planejamento de trajetória também foram revisadas. Essas foram apresentadas dentro de um contexto de deslocamento de múltiplos robôs em mapas conhecidos.

Neste capítulo foi definido o problema de exploração a ser estudado neste trabalho e apresentado o ambiente assumido para testar o método proposto. Foram descritas as características desse ambiente que é caracterizado por ser *indoor* e estruturado, composto por estruturas como corredores, esquinas e cruzamentos. É assumido que o ambiente é formado apenas por paredes paralelas e ortogonais.

# 3 CONSTRUÇÃO DE MAPAS

A navegação baseada em mapas é um processo bastante natural para o ser humano, uma vez que os mapas apresentam uma descrição bastante conveniente do ambiente e podem ser facilmente compartilhados com outras pessoas. No entanto, um nível de processos cognitivos bastante elevados é necessário para interpretar e estabelecer uma correspondência entre os dados apresentados em um mapa com o ambiente que cerca o indivíduo. Frequentemente, o estabelecimento dessa correspondência é facilitado através de modificações no ambiente, como é o caso da adição de placas identificando ruas e estações de metrô [Filliat e Meyer 2003].

Quando o indivíduo em questão é um de robô móvel autônomo, essas capacidades cognitivas são limitadas. Além disso, a necessidade de modificação do ambiente inviabiliza a utilização de robôs em diversas aplicações. O desafio consiste, então, em propor esquemas de navegação baseados em mapas que não pressuponham processos cognitivos de alto nível e que sejam capazes de lidar com ambientes não modificados.

De acordo com [Stachniss 2009] a atividade de construção do mapa de um ambiente desconhecido requer a solução de três etapas, que são *mapeamento*, *localização* e *planejamento de trajetória*. O mapeamento refere-se ao processo de interpretação das informações sensoriais adquiridas e da integração destas em uma representação coerente do ambiente. Já a localização é o processo de estimação da posição do robô em relação a um mapa. Por fim, o planejamento de trajetória é o processo de obtenção de uma série de ações para alcançar de forma eficiente uma posição desejada no ambiente.

Essas três etapas não podem ser resolvidas de forma independente. Antes de interpretar uma série de dados sensoriais para construção de um mapa, o robô necessita da informação do lugar onde esses dados foram obtidos. Por outro lado, a localização do robô dentro de um mapa requer que esse mapa exista. O planejamento de uma trajetória para alcançar uma posição desejada também é dependente das etapas anteriores. Para planejar uma trajetória é necessário o conhecimento da posição inicial do robô, assim como da aparência geral do ambiente para que se possa calcular um caminho entre esses dois pontos.

A Figura 3 ilustra as etapas de mapeamento, localização e planejamento de trajetória, assim como a combinação desses aspectos.

A *localização e mapeamento simultâneos* (*simultaneous localiza-*

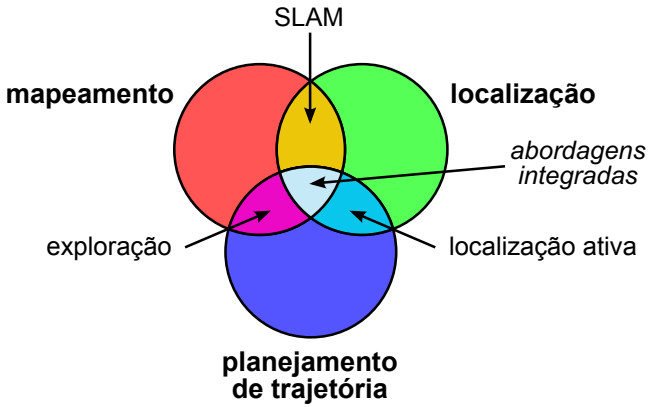



Figura 3: Etapas envolvidas no problema de construção de mapas *[Stachniss 2009]*

*tion and mapping - SLAM*) aborda o problema da construção de um mapa do ambiente enquanto simultaneamente localiza o robô dentro desse mapa. Neste problema as etapas de mapeamento e localização estão acopladas: é necessária uma representação do ambiente para poder estimar a localização do robô dentro desse mapa enquanto uma boa estimativa da posição é necessária para a construção do mapa.

A *localização ativa* procura guiar o robô para locais específicos do mapa com intuito de melhorar a estimativa da posição do robô. Essa abordagem assume a existência de um modelo do ambiente e o utiliza para melhorar o processo de localização.

Já a abordagem de *exploração*, foco desse trabalho, se concentra em guiar o robô de forma eficiente pelo ambiente, com intuito de construir um mapa. Nessa abordagem a localização do robô é assumida como conhecida e precisa. Um exemplo da ausência da necessidade de trabalhar aspectos relacionados à localização é a exploração de ambientes com as características apresentadas na Seção 2.4. Neste caso, as próprias ações previstas pelo planejador de trajetória são suficientes para atualizar informações de localização do robô. Caso o planejador de trajetória envie um comando "vire à direita" ao robô, ele assume que o robô rotacionou 90 graus, cabendo ao próprio robô a correção de eventuais erros. A correta interpretação e execução desses comandos são, então, resolvidos em uma etapa de "navegação", que na abordagem deste trabalho garante a correta movimentação dos robôs pelo ambi-

ente.

Nas abordagens integradas a construção de um modelo do ambiente é realizada por um robô móvel adquirindo informações dos sensores e autonomamente se deslocando através do ambiente. Sempre que o robô se move, ele considera ações que melhoram a sua localização, que favoreçam a aquisição de informações sobre regiões desconhecidas, e que forneçam melhorias para o mapa revisitando regiões as quais se tem incertezas associadas. No final desse processo, o resultado esperado é a obtenção de um modelo completo do ambiente e a determinação da localização do robô dentro desse modelo [Stachniss 2009].

## 3.1 MAPEAMENTO

O problema de mapeamento na robótica refere-se ao processo de interpretação das informações sensoriais adquiridas e da integração delas em uma representação coerente do ambiente. É nessa etapa que o mapa é efetivamente construído, utilizando-se da localização e do planejamento de trajetória para fazê-lo.

As fontes de informações sensoriais que são utilizadas para a navegação e para o mapeamento de um ambiente podem ser classificadas como de origem interna e externa [Filliat e Meyer 2003]. As fontes internas são aquelas ligadas a informações de movimentação do robô, como aceleração, velocidade ou número de rotações realizadas pelas rodas. As fontes externas são aquelas ligadas à percepção do ambiente, como imagens obtidas com câmeras, distâncias medidas com sonares ou lasers, concentração de gás, etc.

A integração dos dados fornecidos pelas fontes internas de informação é suficiente para estimar a posição do robô durante a sua movimentação no ambiente. Esta, porém, apresenta erro acumulativo fazendo com que medições realizadas por longos períodos de tempo tenham sua "qualidade" continuamente reduzida, chegando ao ponto que a estimativa de posição não seja mais confiável. As fontes de informação externas, por sua vez, não apresentam erro acumulativo, porém, podem apresentar ambiguidade na percepção, assumindo que lugares diferentes sejam os mesmos já visitados. As vantagens e desvantagens dessas duas fontes de informação sensoriais são complementares, fazendo com que a combinação de ambas seja fundamental para que o processo de mapeamento ocorra de forma confiável.

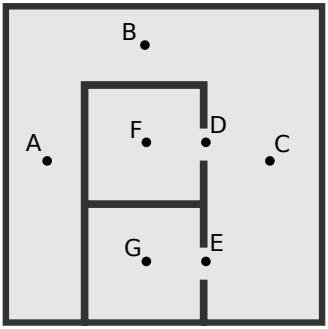
Dadas as duas fontes de dados sensoriais apresentadas, existe uma série de técnicas capazes de integrar essas informações transfor-

mando-as em uma representação coerente do ambiente. Duas categorias principais são tipicamente utilizadas para classificar essas técnicas de modelagem do ambiente: a *abordagem métrica* e a *abordagem topológica*.

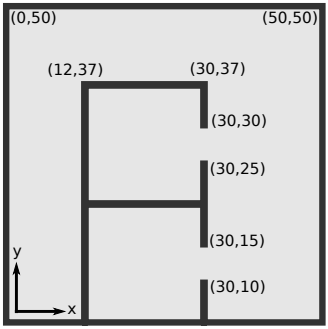
Em *mapas métricos*, o ambiente é representado por um conjunto de estruturas armazenadas pelo robô dentro de um mesmo sistema de referência, descrevendo o ambiente através das informações lidas pelos sensores. Estruturas facilmente extraídas são utilizadas para descrever formatos de objetos e obstáculos inseridos no ambiente, como linhas e pontos, associados às suas respectivas posições. Uma técnica bastante utilizada dentro do contexto de mapas métricos são as grades de ocupação (*occupancy grids*), onde, ao invés de utilizar estruturas para descrever o ambiente, é feita uma discretização do mesmo utilizando uma grade de resolução fixa. A cada uma das células dessa grade é atribuído um valor de probabilidade dela estar sendo ocupada por um obstáculo. Essa técnica apresenta a vantagem de lidar bem com a fusão de dados de diferentes sensores, porém apresenta alguns problemas ligados principalmente à granularidade, escalabilidade e extensibilidade [Bailey e Nebot 2001]. Por um lado a resolução da grade de ocupação deve ser alta o suficiente para capturar detalhes do ambiente, por outro, quanto maior o número de células maior o custo computacional para trabalhar com o mapa.

Os *mapas topológicos* descrevem o ambiente de forma mais abstrata, identificando pontos específicos, como estruturas do ambiente (portas, corredores) ou *landmarks*. O resultado do mapeamento com essa abordagem é um grafo, onde os nós são as entidades identificadas e os arcos são as relações de acessibilidade entre esses nós, ou seja, a existência de adjacência entre dois nós significa que no ambiente existe uma passagem entre os mesmos. Na Figura 4 são apresentadas as principais diferenças entre as duas abordagens de representação de mapa. Na Figura 4(a) é mostrado ambiente real; na Figura 4(b) o mapa métrico, onde a posição dos objetos é medida e suas coordenadas guardadas dentro de um sistema de coordenadas comum. A distância entre objetos e pontos do mapa pode ser facilmente deduzida a partir de suas coordenadas. Na Figura 4(c) é apresentada a representação topológica, na qual somente pontos com características específicas são guardados, juntamente com a sua relação espacial.

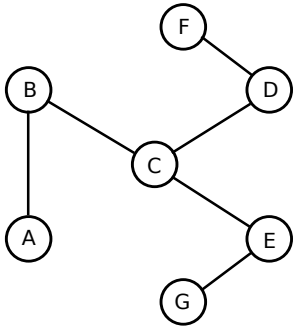
Em [Thrun 1998] são apresentadas as principais vantagens e desvantagens de cada abordagem. Mapas métricos, em geral, são de fácil construção uma vez que os dados lidos pelos sensores são diretamente incorporados ao mapa; essa abordagem também não possui problemas




(a) Ambiente real




(b) Mapa métrico




(c) Mapa topológico

Figura 4: Abordagens de representação de mapas

de ambiguidade no reconhecimento de lugares, já que a os dados ambientais são armazenados juntamente com sua respectiva posição dentro de um único sistema de referência.

Por outro lado, mapas métricos possuem alta complexidade em questão de armazenamento e processamento, já que a resolução do mapa não depende da complexidade do ambiente. Nesse ponto a abordagem de mapas topológicos se destaca: ela é extremamente compacta, o que lhe confere vantagens, como o rápido processamento de cálculos de trajetórias e a facilidade de interação com planejadores simbólicos.

Outro ponto em que as duas abordagens diferem é na questão de necessitar de uma determinação precisa da posição do robô. Enquanto a abordagem métrica é bastante dependente devido ao fato de precisar atribuir uma posição aos elementos encontrados no ambiente, a abordagem topológica consegue contornar melhor problemas causados por

efeitos como derrapagens e deslizamentos (*drift/slippage*).

Como as abordagens métricas e topológicas possuem vantagens complementares, uma série de estudos tem sido realizados utilizando abordagens híbridas. Essas abordagens buscam integrar aspectos métricos e topológicos para evitar os pontos fracos de cada abordagem e combinar suas vantagens. Essa integração geralmente é tratada através da incorporação de dados métricos em mapas topológicos [Marjovi e Marques 2011] ou através da extração de um mapa topológico de uma representação métrica [Tomatis, Nourbakhsh e Siegwart 2003].

A primeira abordagem consiste na construção de um mapa topológico e na inclusão de dados geométricos em cada nó do mapa, com informações de onde este nó está inserido no ambiente. Essas informações facilitam principalmente a resolução do problema de determinação de equivalência. Já a segunda abordagem consiste em, basicamente, dividir um mapa métrico em regiões significantes. Essas regiões métricas passam a ser nós de um mapa topológico global que mantém informações da conectividade entre as mesmas. Essa abordagem reduz o volume de dados com os quais tem que se trabalhar para realizar de um planejamento de trajetória, por exemplo.

## 3.2 PLANEJAMENTO DE TRAJETÓRIA

Com a disponibilização de um mapa (parcial ou total) e da localização do robô dentro desse mapa, o próximo passo é a execução da movimentação dentro desse ambiente. Essa movimentação deve possibilitar que, a partir de uma posição inicial, o robô possa atingir uma posição final desejada de forma eficiente e segura. O planejamento de trajetória é a etapa que realiza o cálculo do caminho entre essas posições.

Existem duas formas principais de planejamento de trajetória: o *planejamento sobre um espaço discretizado* e o *planejamento sobre um espaço contínuo* [Meyer e Filliat 2003]. O planejamento sobre o espaço discretizado é amplamente mais utilizado no contexto da robótica móvel, principalmente quando o objetivo é a coordenação de um time de robôs buscando a realização de um objetivo global. Neste caso, a representação topológica do mapa apresenta uma vantagem sobre a representação métrica: o planejamento é feito diretamente sobre o mapa através de técnicas de busca em grafo. Em geral, durante o mapeamento topológico, os elementos selecionados como nós do grafo são, já nessa etapa, adequados às técnicas de planejamento.

Mesmo no caso do mapeamento através da abordagem métrica, o resultado pode ser a obtenção de um mapa representado como um espaço discreto. É o caso da técnica das *grades de ocupação*, apresentada na Seção 3.1. Nessa técnica uma grade igualmente espaçada é colocada sobre o ambiente e cada célula é preenchida com a probabilidade de existir um obstáculo naquele ponto. Neste caso, o planejamento sobre um espaço discretizado também pode ser diretamente utilizado. Esta técnica, entretanto, pode apresentar problemas como o elevado número de estados discretos, dificultando o planejamento.

Quando mapas são construídos através da abordagem métrica e não resultam num espaço discretizado, uma etapa anterior ao planejamento geralmente consiste em discretizá-los. Com o mapa discretizado as mesmas técnicas de busca em grafo utilizadas nos mapas topológicos podem ser aplicadas. As técnicas *decomposição em células* e *roadmaps* de discretização de mapas são apresentadas na Seção 3.3.

É possível, entretanto, realizar o planejamento diretamente sobre o espaço contínuo. O número de técnicas existentes, neste caso, é limitado, assim como a sua aplicação. Essas técnicas, em geral, calculam a próxima ação do robô a cada mudança de posição, dificultando a realização de um planejamento para múltiplos robôs garantindo a ausência de colisões e bloqueios. A seção 3.2.1 apresenta o método de *campos potenciais* para planejamento diretamente em um espaço contínuo.

### 3.2.1 Planejamento em espaço contínuo

Quando mapas métricos são utilizados, é possível planejar diretamente no espaço contínuo, sem recorrer a discretização. Ao invés disso, uma função que indica para onde se mover para alcançar o objetivo é calculada em cada ponto do espaço livre [Meyer e Filliat 2003].

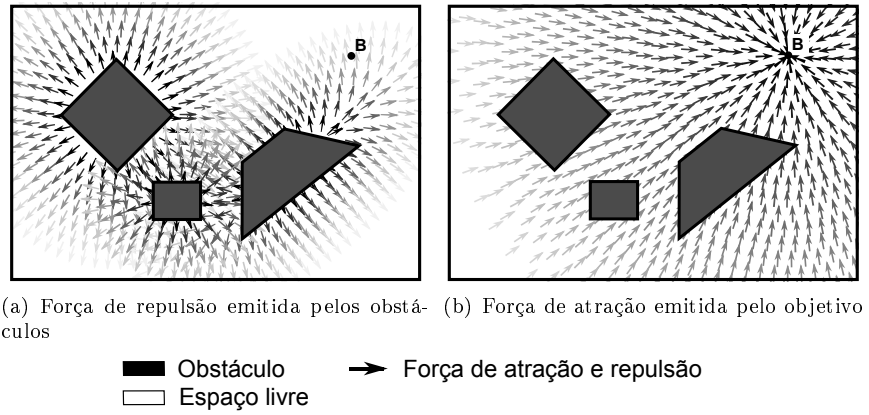
A técnica mais difundida nesse contexto é a dos *campos potenciais* [Latombe 1990]. A função calculada através desse método consiste em um componente atrativo, gerado pelo objetivo, e em um componente repulsivo, gerado pelos obstáculos. O valor resultante da soma desses dois componentes representa a "força" aplicada ao robô em cada instante de tempo, que deverá afastá-lo de obstáculos levando-o ao seu objetivo.

Esse método não necessita da criação de nenhuma estrutura de dados prévia, como a construção de um grafo. Isso possibilita sua utilização em ambientes com características dinâmicas. Devido a não necessidade da criação prévia de estruturas de dado, nem de um cál-

culo global de planejamento, o método se destaca também pela baixa complexidade. No entanto, o método está sujeito a alguns problemas, como os mínimos locais, no qual o robô pode ficar preso em um ponto do ambiente sem alcançar o objetivo. Outro problema é que esta técnica pode apresentar movimentos oscilatórios na presença de obstáculos específicos.

Alguns trabalhos foram realizados a fim de buscar contornar os problemas inerentes a essa técnica, porém apresentam, em geral, soluções computacionalmente complexas. Dessa forma, os campos potenciais são geralmente aplicados em ambientes pequenos, ou em abordagens híbridas. Nessas abordagens o planejamento global é realizado sobre o mapa discretizado e problemas locais, como o desvio de obstáculos, podem ser resolvidos com essa técnica.

Na Figura 5 é apresentado um exemplo de um planejamento de trajetória feito com campos potenciais. Na Figura 5(a) são mostradas as forças de repulsão emitidas pelos obstáculos do ambiente e na Figura 5(b) as forças de atração do objetivo final do robô. A soma vetorial das duas forças determinará para onde o robô deverá se mover.




(a) Força de repulsão emitida pelos obstáculos (b) Força de atração emitida pelo objetivo

Figura 5: Campo potencial sobre ambiente com obstáculos

### 3.2.2 Planejamento em espaço discreto

Quando o planejamento de trajetória é realizado sobre um espaço discretizado ele é, em geral, realizado em dois níveis distintos de planejamento. O primeiro deles atua num nível discreto e mais alto e é

responsável pela geração de uma lista de instruções de movimentação a serem executadas sequencialmente pelo robô para alcançar o seu objetivo. Cada uma das instruções executadas representa uma mudança na posição discreta do robô dentro mapa. O segundo nível de planejamento atua de forma contínua e é responsável por traçar um caminho entre as duas posições discretas em questão. Ele controla a execução de cada uma das instruções do nível anterior e é responsável por corrigir eventuais "ruídos" na movimentação do robô. Muitas vezes esse segundo nível atua também de forma reativa, esperando a passagem de outro robô ou contornando pequenos obstáculos não existes no mapa, por exemplo, sem a necessidade de mudanças nos níveis mais altos do planejamento de trajetória.

Na Figura 6 é mostrado o trajeto de um robô em um mapa discretizado. Assumindo que o planejamento de trajetória seja realizado em dois níveis, o nível discreto do planejamento pode ser responsável, por exemplo, pelo cálculo da sequência de células a serem percorridas para que o robô chegue, a partir da célula 1, à célula número 18. A sequência de células, de acordo com a Figura 6 é $1 \rightarrow 2 \rightarrow 7 \rightarrow 12 \rightarrow 17 \rightarrow 18$. Neste caso, o nível contínuo do planejamento seria responsável por calcular o trajeto para o deslocamento entre duas células previstas no nível anterior.

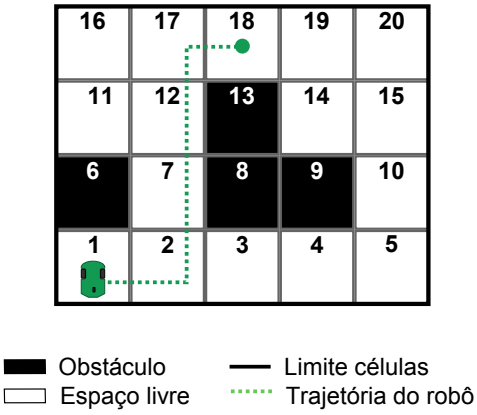



Figura 6: Exemplo de planejamento de trajetória sobre um espaço discretizado

## 3.3 DISCRETIZAÇÃO DE MAPAS

Técnicas de discretização de mapas são utilizadas para obtenção de uma representação discreta do ambiente, na forma de um grafo. Essas técnicas são aplicadas sobre mapas métricos, total ou parcialmente conhecidos, e o grafo resultante é utilizado para realização de um planejamento de trajetória dentro da abordagem discreta.

Em geral a discretização ocorre posteriormente ao mapeamento, sendo por vezes considerada uma das etapas do planejamento de trajetória. Muitas vezes, porém, a discretização e o mapeamento são feitos de forma simultânea. Neste caso, logo após o mapeamento de uma porção do ambiente, características são extraídas e um novo nó é incorporado ao grafo.

Nessa seção são apresentados alguns métodos de discretização de mapas métricos, dos quais algumas características podem ser extrapoladas para o processo de mapeamento topológico. Mais técnicas, assim como uma descrição mais detalhada das mesmas podem ser encontradas em [Latombe 1990] e [Choset et al. 2005].

### 3.3.1 Decomposição em células

O primeiro método de discretização do espaço de busca consiste na divisão desse espaço em pequenas regiões denominadas células. Essas células irão compor um grafo não direcionado de representação do ambiente, no qual as mesmas são os nós. Células que possuem região de fronteira compartilhada são consideradas células adjacentes, e a relação de adjacência entre essas células é representada no grafo pelos arcos conectando os nós. Esse grafo resultante é chamado *grafo de conectividade*.

Durante a divisão do ambiente, as células devem ser moldadas de forma que um caminho possa ser facilmente traçado entre duas posições quaisquer de um robô entre células adjacentes, devendo, então, apresentar geometria simples e a não sobreposição entre duas células. Outro ponto é que se deve ter uma ponderação em relação ao tamanho das células: excessivamente grandes elas não representam o ambiente e muito pequenas geram uma complexidade desnecessária no cálculo de caminhos. Em um caso extremo podemos ver o ambiente todo classificado como uma única célula; neste caso voltamos ao problema original.

Com o grafo construído pode-se executar uma busca no mesmo para encontrar uma sequencia de células a ser percorrida de forma que,

a partir da célula contendo o ponto inicial do robô, ele possa atingir a célula contendo a posição final desejada.

Os métodos baseados em decomposição em células podem ser divididos em *exatos* e *aproximados*. Métodos exatos de decomposição dividem apenas o espaço livre de obstáculos do mapa, sendo a união das células exatamente igual ao espaço livre. Nos métodos baseados em decomposição aproximada, o espaço livre não é necessariamente inteiramente divido, em alguns casos apenas regiões de interesse passam a constituir as células. Nesses métodos as células resultantes do processo de decomposição estão contidas no espaço livre do mapa.

#### 3.3.1.1 Decomposição exata

*Decomposição trapezoidal*

A decomposição trapezoidal [Latombe 1990] é o exemplo mais comum de decomposição celular exata. Ela é aplicada a ambientes nos quais os obstáculos estão limitados a polígonos e é realizada através de uma varredura do ambiente por uma linha (horizontal ou vertical). Sempre que esta linha intercepta um vértice de um obstáculo ela cria novos limites de células. Nessa técnica as células geradas são sempre de formato trapezoidal ou triangular, tendendo a serem finas e compridas; características negativas para a correção de posicionamento e para a realização de tarefas envolvendo múltiplos robôs. Na Figura 7 é mostrado o processo de decomposição em células pelo método de decomposição trapezoidal. Na Figura 7(a) é mostrado o mapa ainda não discretizado; nas Figuras 7(b) e 7(c) o processo de discretização e, por fim, na Figura 7(d) é mostrado o mapa divido em células.

*Decomposição por arranjo*

A decomposição por arranjo [Sleumer e Tschichold-Gürman 1999] é também classificada como uma técnica de decomposição celular exata, aplicada a ambientes com obstáculos poligonais. Nessa técnica, as arestas pertencentes a obstáculos são estendidas até alcançarem algum outro objeto ou limite do ambiente. Esse arranjo formado pelas linhas estendidas é utilizado para a formação das células.

Uma interessante propriedade é que as células geradas são sempre polígonos convexos, assim como a união de duas células adjacentes. Isso permite que, quando geradas células muito pequenas ou quando realizadas divisões desnecessárias de uma área, uma das etapas da de-

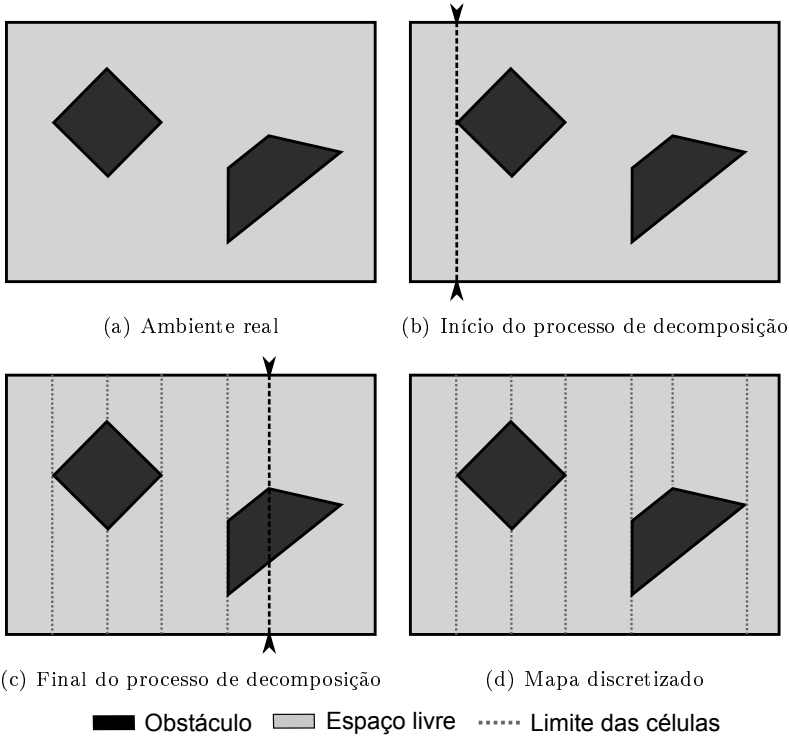



Figura 7: Decomposição celular pelo método trapezoidal

composição realize a fusão de células, produzindo um resultado mais interessante para o planejamento de trajetórias. O número de células geradas está diretamente ligado à complexidade do ambiente; quanto mais obstáculos e quanto mais arestas eles possuírem, maior o número de células e maior detalhamento do ambiente. Na Figura 8 é apresentado o processo de decomposição por arranjo. Na Figura 8(a) é mostrado o mapa ainda não discretizado; na Figura 8(b) são estendidas as arestas dos obstáculos para formação das células; na Figura 8(c) as células classificadas como muito pequenas são fundidas com células adjacentes e na Figura 8(d) é mostrado o resultado da decomposição por arranjo.

*Decomposição por triangulação*

A divisão do espaço de trabalho em células triangulares é bas-

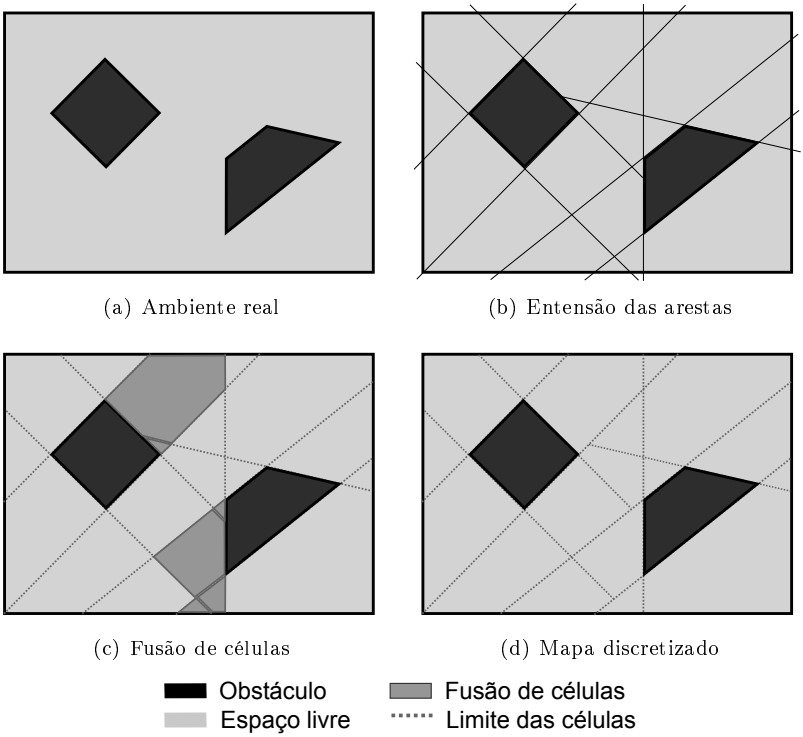



Figura 8: Decomposição celular pelo método de arranjo

tante utilizada por apresentar uma série de algoritmos computacionalmente eficientes, representação algébrica simples e por serem capazes de decompor ambientes com obstáculos de formatos variados. Em geral utilizam características geométricas do ambiente, como vértices, e arestas, criando mais células quando o ambiente é mais complexo. Esse método é também classificado como decomposição exata.

Existem diversos métodos e algoritmos capazes de realizar triangulação para o planejamento de trajetória, cada um trazendo diferentes características para a formação das células, como pode ser visto em [Gong e Wang 2008]. O método mais difundido no contexto da robótica móvel é a triangulação pelo método Delaunay, onde se busca maximizar o menor ângulo dos triângulos gerados, tendendo a evitar triângulos de formato alongado; propriedade importante para a decomposição celular no planejamento de trajetória (mais detalhes sobre o

método em [Berg et al. 2008]). A triangulação pelo método Delaunay é aplicada sobre um conjunto de pontos dados, utilizados como vértices dos triângulos. A decomposição ocorre formando triângulos de forma que, ao ser traçadas circunferências passando por todos os vértices de um triângulo, nenhum ponto fique no interior das mesmas. A Figura 9 exemplifica o resultado desse processo.

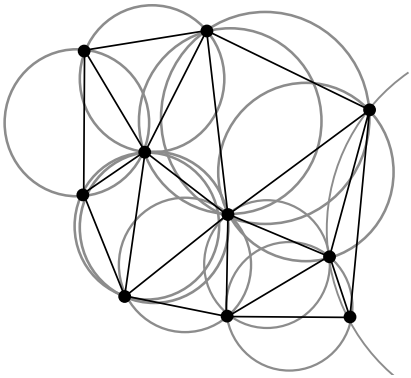


Figura 9: Exemplo de triangulação de Delaunay

Para ambientes com obstáculos, como os que foram tratados nas seções anteriores, é utilizada a técnica de Triangulação de Delaunay com Restrições (*Constrained Delaunay triangulation - CDT*), uma modificação da técnica original que força a presença de alguns segmentos na triangulação, neste caso, as arestas pertencentes a obstáculos. Como a triangulação de Delaunay geralmente possui solução única, a imposição de segmentos pode levar a não satisfação das condições, o que é aceitável nessa técnica. Na Figura 10 é mostrada a decomposição de um ambiente com obstáculos utilizando a técnica de triangulação de Delaunay com restrições.

#### 3.3.1.2 Decomposição aproximada

*Grades de ocupação fixa*

As grades de ocupação (*occupancy grids*), como apresentado anteriormente, são uma forma de discretização e armazenamento de mapas métricos. Muitas vezes, porém, essa discretização só ocorre na etapa de planejamento de trajetórias.

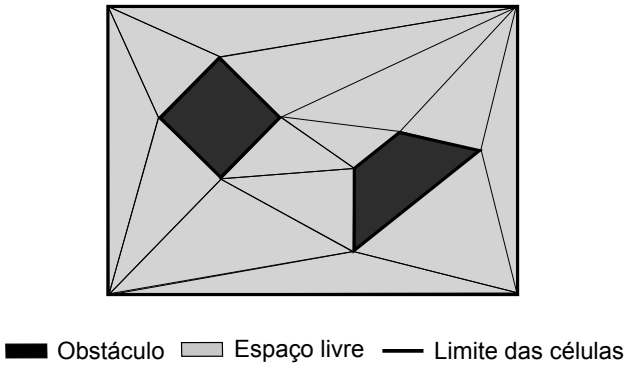
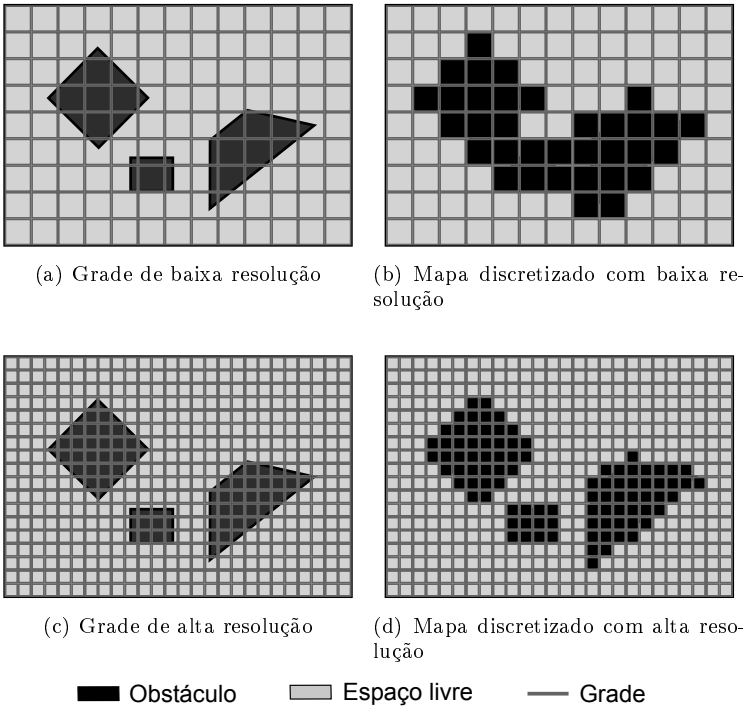



Figura 10: Decomposição celular pelo método de triangulação de Delaunay com restrições

Esse é o método mais difundido no planejamento de trajetória, no contexto da robótica móvel. O processo é realizado colocando-se sobre o mapa uma grade igualmente espaçada, de resolução arbitrária. Cada uma das células dessa grade é então analisada: se estiver completamente livre é marcada como "livre" e caso abranja qualquer porção de obstáculo é marcada como "ocupada". As células livres compõe, então, o grafo onde serão utilizadas técnicas de busca. Como a união das células livres está contida no espaço livre do mapa, porém não é necessariamente igual a ele, o método é classificado como decomposição aproximada.

A eficiência desse método está diretamente ligada ao ajuste da resolução da grade que será utilizada para decompor o ambiente em células. Células de tamanho grande reduzem o volume de recursos computacionais a serem utilizados, acelerando a busca, porém podem causar uma perda de precisão indesejada, fazendo com que caminhos não sejam encontrados mesmo quando eles existem. Por outro lado, células de tamanho muito pequeno capturam mais detalhes do ambiente, porém podem inviabilizar a busca no grafo pela grande quantidade de nós. Nas Figuras 11(a) e 11(b) é mostrado um exemplo de discretização utilizando uma grade de ocupação de resolução baixa. O baixo número de células faz com detalhes do mapa sejam perdidos, como os caminhos existentes entre os obstáculos. Nas Figuras 11(c) e 11(d) a resolução da grade é aumentada e consequentemente o número de células. Nessa discretização é possível notar que o nível de detalhamento do mapa discretizado é maior.




(a) Grade de baixa resolução

(b) Mapa discretizado com baixa resolução

(c) Grade de alta resolução

(d) Mapa discretizado com alta resolução

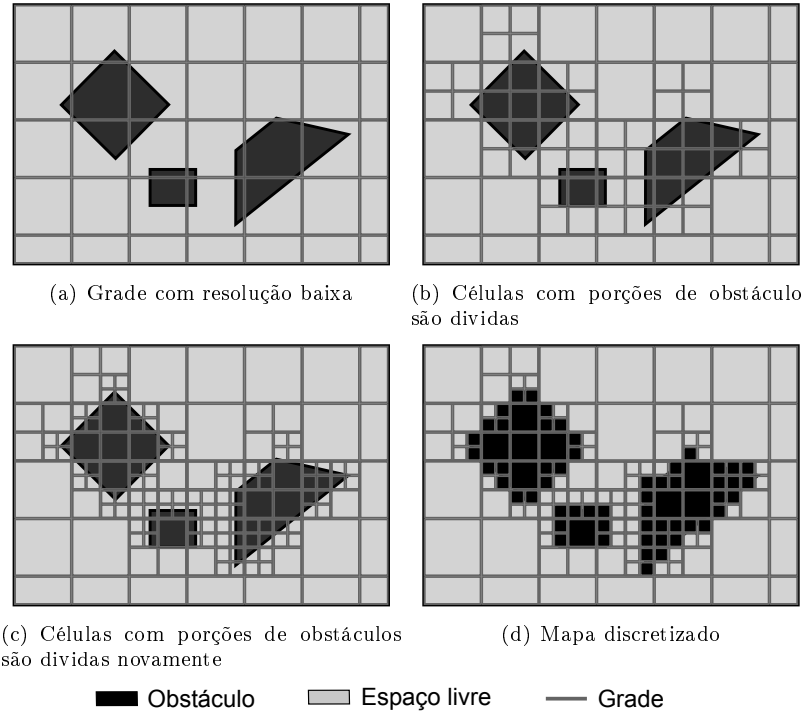
Figura 11: Discretização utilizando grades de ocupação fixa

*Grades de ocupação variável*

Uma extensão do método de grades de ocupação clássico é o método de grades de ocupação variável, onde são utilizadas células de variados tamanhos. Essa técnica visa contornar o problema existente no método anterior da relação entre captura de detalhes do mapa e desempenho da busca no grafo, causados pela utilização de uma resolução fixa da grade. Um exemplo de sua utilização pode ser encontrado em [Kaplow, Atrash e Pineau 2010].

Nesse método, o ambiente é inicialmente decomposto com uma grade de resolução baixa (células grandes). As células resultantes são então avaliadas: aquelas classificadas como livres (sem nenhuma porção de obstáculo) não serão alteradas, aquelas que estão totalmente imersas em obstáculos também não serão alteradas e aquelas que abrangem apenas uma porção de um obstáculo são novamente divididas e clas-

sificadas. O processo se repete várias vezes e o resultado é uma decomposição com mais detalhes nas proximidades de obstáculos e com um menor número de células se comparado com o método de resolução fixa. Na Figura 12 é mostrado um exemplo de decomposição com grades de ocupação variável. Na Figura 12(a) o mapa é divido por uma grade de resolução baixa; nas Figuras 12(b) e 12(c) as células "mistas" são novamente dividas e reavaliadas. Na Figura 12(d) é apresentado o resultado da discretização.




(a) Grade com resolução baixa

(b) Células com porções de obstáculos são dividas

(c) Células com porções de obstáculos são dividas novamente

(d) Mapa discretizado

Figura 12: Discretização utilizando grades de ocupação variável

### 3.3.2 Roadmaps

Essa abordagem de discretização do mapa consiste em capturar a conectividade do espaço livre do ambiente em uma rede de curvas de uma dimensão, denominada *roadmap* [González-Banos, Hsu e Latombe

2006]. Ao invés de decompor o ambiente em áreas discretas, ele é dividido em uma série de possíveis caminhos. Uma vez que esta rede tenha sido construída o robô está restrito a mover-se somente ao longo dessas curvas. O planejamento de trajetórias reduz-se, então, a conectar as posições inicial e final do robô ao *roadmap* e buscar neste um caminho entre estes dois pontos.

Em geral, os métodos de *roadmap* são rápidos e simples de implementar e de baixo custo de armazenamento. Como os *roadmpas* impõe restrições às possibilidades de caminhos dos robôs o desafio consiste em empregar técnicas que construam essa rede de curvas de forma que só exista um caminho livre de colisões entre dois pontos no ambiente se existir um caminho livre de colisões formado pelas curvas. Duas técnicas clássicas que atendem a esses requisitos são os *grafos de visibilidade* e os *diagramas de Voronoi*.

#### 3.3.2.1 Grafos de visibilidade

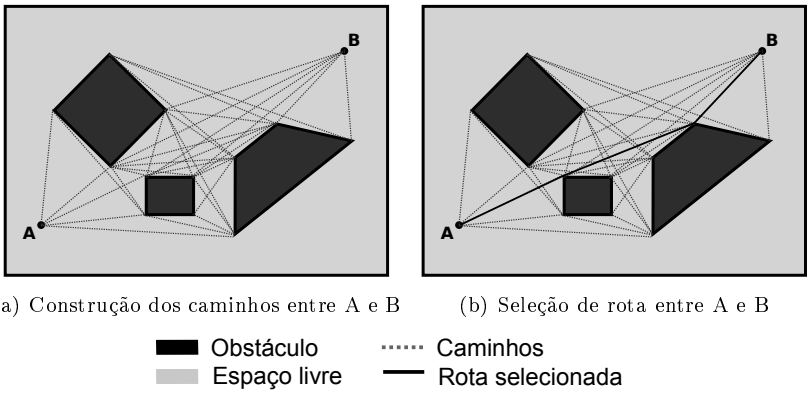
No método do grafo de visibilidade os vértices dos polígonos do ambiente são considerados nós do grafo. Cada vértice é conectado a todos os outros vértices "visíveis", ou seja, todos os vértices que podem ser conectados através de uma reta sem cruzar nenhum obstáculo. No entanto, essa proximidade excessiva aos obstáculos pode não ser desejável, como nos casos onde o robô se move rápido, devido aos elevados riscos de colisão.

Na Figura 13 é mostrado um exemplo de construção de *roadmap* com o método de grafos de visibilidade. Na Figura 13(a) pode-se ver o *roadmap* construído entre os pontos A e B e na Figura 13(b) é selecionada uma rota entre os mesmos pontos.

#### 3.3.2.2 Diagrama de Voronoi

O Diagrama de Voronoi é um método completo de *roadmap* que tende a maximizar a distância entre o robô e os obstáculos no mapa. Para cada ponto livre no mapa é calculada a distância para o obstáculo mais próximo. O diagrama de Voronoi consiste, então, em curvas conectando os pontos equidistantes a um ou mais obstáculos.

Como o algoritmo maximiza a distância entre o robô e um objeto no ambiente, qualquer sensor de curto alcance no robô poderá falhar para perceber o mundo ao seu redor. Se o sensor está sendo usado para

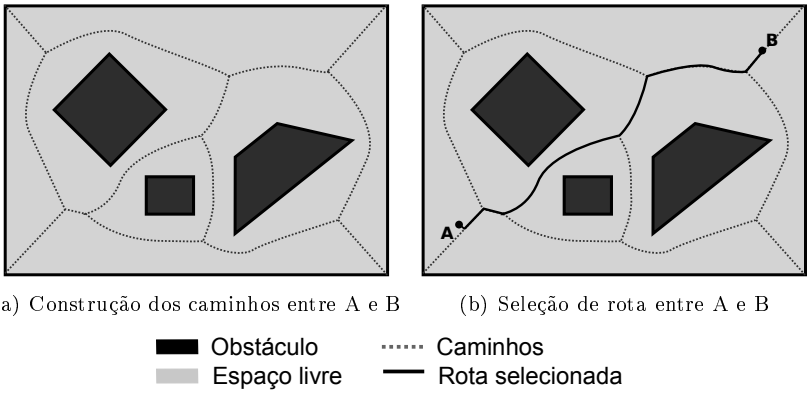



(a) Construção dos caminhos entre A e B (b) Seleção de rota entre A e B

Figura 13: Construção de *roadmap* através de grafos de visibilidade

localização, então, o caminho designado pelo diagrama de Voronoi será pobre do ponto de vista de localização. Por outro lado, o caminho é criado baseado em um ponto equidistante dos obstáculos, garantindo uma rota segura para o robô e mostrando-se uma alternativa aos grafos de visibilidade.

Na Figura 14(a) é mostrada o *roadmap* construído sobre o mapa de um ambiente com obstáculos, utilizando a técnica de diagrama de Voronoi. Na Figura 14(b) uma das rotas do *roadmap* é selecionada para ligar os pontos A e B.




(a) Construção dos caminhos entre A e B (b) Seleção de rota entre A e B

Figura 14: Construção de *roadmap* através de diagrama de Voronoi

## 3.4 CONCLUSÃO

Neste capítulo foram tratados os principais aspectos referentes à construção de mapas de um ambiente desconhecido. Esta atividade, segundo [Stachniss 2009] requer a solução de três etapas: *mapeamento*, *localização* e *planejamento de trajetória*. A combinação dessas etapas gera abordagens específicas para tratar do problema da construção de mapas. A exploração, foco desse trabalho, é gerada a partir da combinação das etapas de mapeamento e planejamento de trajetória. Nessa abordagem, a localização do robô é considerada conhecida e precisa, por esse motivo aspectos referentes a essa etapa não são tratados.

Em relação à etapa de mapeamento existem duas abordagens principais: abordagem métrica e a abordagem topológica. Em mapas métricos, o ambiente é representado por um conjunto de estruturas facilmente extraídas pelos robôs para descrição do formato de objetos e obstáculos, como linhas e pontos. Essas estruturas são armazenadas dentro de um mesmo sistema de referência. Já os mapas topológicos descrevem o ambiente de forma mais abstrata, capturando estruturas específicas do ambiente, como portas ou *landmarks*. O resultando do mapeamento topológico é um grafo, onde são representadas as relações de acessibilidade entre essas estruturas capturadas do ambiente.

O planejamento de trajetória também se divide em duas categorias principais: o planejamento sobre um espaço discretizado e o planejamento sobre um espaço contínuo. O planejamento sobre um espaço discretizado é amplamente utilizado no contexto da robótica móvel, porém existentes técnicas que realizam o planejamento de trajetória contínuo diretamente sobre um espaço métrico, como os campos potenciais. Por outro lado, quando se deseja utilizar uma abordagem de planejamento sobre um espaço discretizado, uma etapa anterior ao planejamento consiste na discretização do mapa.

Uma série de técnicas pode ser utilizada para a realização da discretização de mapas. Neste capítulo foram apresentadas técnicas de decomposição em células, classificadas em exatas e aproximadas, e a técnica de *roadmaps*.

O capítulo seguinte trata do método de exploração de ambientes desconhecidos proposto nesse trabalho, apresentando as abordagens selecionadas para o mapeamento e o planejamento de trajetória. São apresentados o algoritmo para coordenação dos múltiplos robôs, no contexto do planejamento de trajetória, e heurísticas utilizadas na alocação de tarefas para aumento da velocidade de exploração.

# 4 MÉTODO PROPOSTO

Conforme apresentado nos capítulos anteriores, a atividade de exploração de ambientes desconhecidos é composta por duas etapas principais. O mapeamento é responsável por interpretar as informações sensoriais e, a partir delas, obter uma representação coerente do ambiente. Já o planejamento de trajetória é responsável por calcular caminhos para os robôs atingirem pontos de interesse no mapa.

As duas etapas apresentam desafios, alguns inerentes ao problema de exploração, outros gerados pela utilização de sistemas multi-robôs para sua resolução. O mapeamento, por exemplo, tem que lidar com desafios relacionados a incertezas sensoriais e com a determinação de equivalência de regiões do mapa. Já no planejamento de trajetória, estão presentes desafios como a coordenação dos múltiplos robôs e a alocação de tarefas.

Nesse capítulo é proposto um método de exploração de ambientes desconhecidos com um sistema multi-robô. As abordagens selecionadas para as etapas de mapeamento e planejamento de trajetória são apresentadas e soluções são propostas para os desafios existentes nas mesmas. Entre os desafios, dois ganham atenção especial por concentrarem as principais contribuições desse trabalho: a alocação de tarefas e a coordenação do sistema multi-robô.

As soluções propostas para esses desafios visam a diminuição do tempo total de exploração do ambiente, tratando aspectos como a diminuição da interferência entre as trajetórias dos robôs (causando esperas e desvios), a diminuição da passagem dos mesmos por regiões já exploradas e o tempo de cálculo das etapas.

## 4.1 ETAPAS DO PROCESSO DE EXPLORAÇÃO

O processo de construção de mapa proposto nesse trabalho é apresentado na Figura 15. O processo é iniciado através da aquisição de dados a partir dos sensores dos robôs de uma região inicialmente inexplorada do ambiente. Esses dados são interpretados pelos robôs e as características do local de onde eles foram coletados são incorporadas ao mapa global do sistema.

Com um região do mapa conhecida é possível determinar se existe a necessidade de executar uma alocação de tarefas, e consequente cálculo de trajetória, para que os robôs atinjam novas regiões inexplo-

radas. Caso o robô já se encontre no limite entre uma região explorada e não explorada basta definir a direção que ele deve prosseguir para continuar o processo de exploração. Essa decisão não interfere nas trajetórias dos demais robôs e, por isso, não necessita de um novo planejamento de trajetórias.

Algumas situações, entretanto, fazem com que os robôs tenham que percorrer regiões já exploradas do ambiente para continuar a exploração. Essas situações, apontadas na Figura 15, são a descoberta de um beco-sem-saída, o encontro entre robôs e o fechamento de um *loop*. Nesses casos, é exigida uma avaliação do lugar mais vantajoso para o robôs se deslocarem, assim como um cálculo de trajetórias que garantam esses deslocamentos de forma segura.

A execução dessa movimentação leva os robôs para novas regiões do ambiente, do qual são extraídos novos dados através dos sensores. Esse processo é repetido até que o ambiente seja completamente explorado.

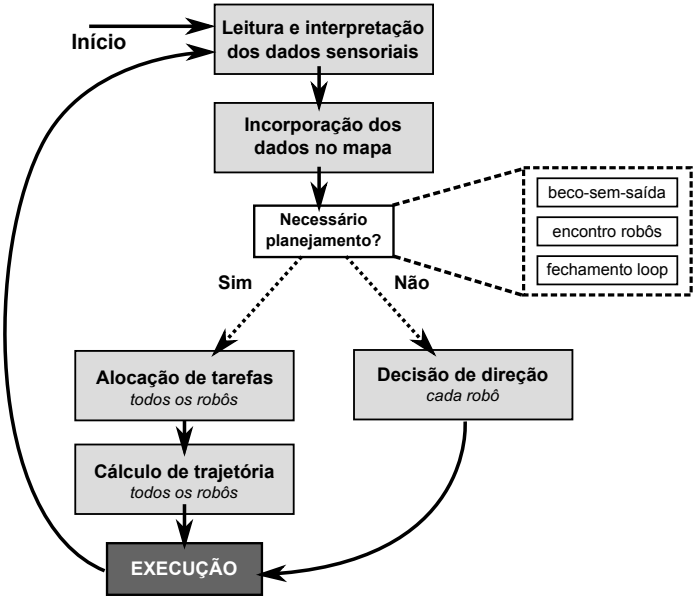



Figura 15: Fluxograma do processo de exploração proposto

A arquitetura utilizada neste trabalho para execução da exploração é centralizada, ou seja, todo processo de alocação de tarefas, cálculo

de trajetórias e construção do mapa do ambiente é realizado em um único agente. Neste trabalho, o agente responsável por essas etapas é um computador central para o qual é assumido que não existem restrições de comunicação com os robôs. Cabe aos robôs do sistema receber e executar comandos discretos enviados pelo coordenador e interpretar dados sensoriais enviando-os para o coordenador.

## 4.2 MAPEAMENTO

Neste trabalho é utilizada uma abordagem topológica para integração das informações sensoriais adquiridas pelos robôs e construção de um mapa que represente o ambiente de forma coerente. Nesse mapa topológico são incorporadas características métricas, facilmente mensuradas durante a exploração. O resultado é a representação do ambiente através de um grafo, representação compacta e de rápido processamento, porém, com dados da localização dos nós inseridos num sistema de coordenadas. Essa abordagem híbrida resultante facilita a resolução de problemas recorrentes na abordagem topológica, como a determinação de equivalência de regiões do ambiente.

Como apresentado na Seção 2.4, o ambiente em questão é *indoor* e estruturado, formado por paredes paralelas e perpendiculares. Para a construção do mapa, é necessário determinar quais estruturas de interesse do ambiente serão identificadas e transformadas nos nós do grafo. O método proposto neste trabalho utiliza o conceito de *features* para tal.

***Features* são estruturas do ambiente que apresentam características padrões, tais como corredores ou esquinas.** Devido ao tipo de ambiente considerado neste trabalho, cinco tipos de *features* podem ser extraídas do ambiente: corredores, esquinas, junções, cruzamentos e becos-sem-saída (*dead ends*), conforme mostrado na Figura 16.

O mapa do ambiente é representado, então, por um grafo não orientado no qual os corredores são os arcos e as demais *features* são os nós. Nós adjacentes representam *features* do ambiente que estão conectadas a um corredor em comum. A cada um dos nós do grafo é associado uma par de coordenadas correspondente a localização aproximada de uma *feature* em relação as demais. Esse par de coordenadas é tratado aqui como a posição do nó.

A Figura 17 mostra a representação em grafo do ambiente apresentado na Figura 16. Neste caso, os nós 3 e 4 são adjacentes, pois as

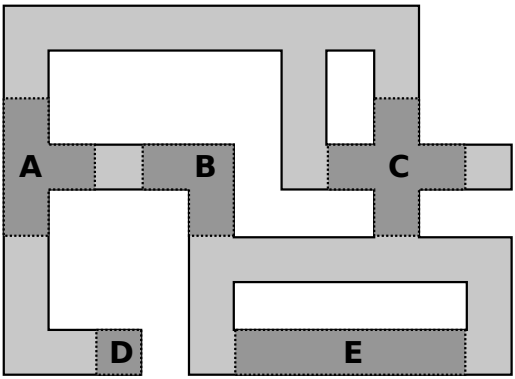



Figura 16: Exemplo de ambiente com identificação de *features*
*A: junção, B: esquina, C: cruzamento, D: beco-sem-saída, E: corredor*

*features* A e B estão conectadas a um mesmo corredor.

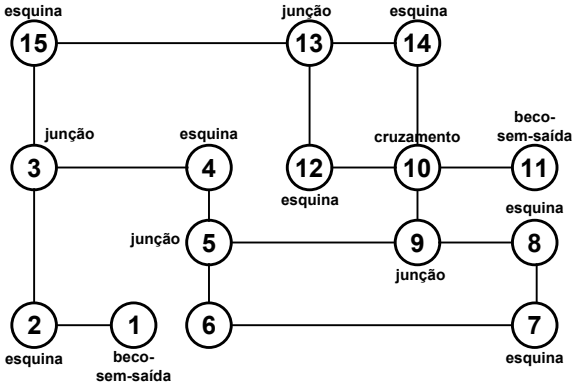



Figura 17: Exemplo de representação em grafo dos ambientes estudados

A abordagem topológica com dados métricos baseada em *features* aqui apresentada é baseada em [Marjovi e Marques 2011]. As seções seguintes, no entanto, que tratam da forma como são identificadas as *features* do ambiente, como são construídos os mapas e como é tratado o problema de determinação de equivalência são contribuições desse trabalho.

### 4.2.1 Identificação de features

Vários trabalhos abordam o reconhecimento de *features* através de um "quadro de sensoriamento local", construído com as informações fornecidas pelos sensores de distância disponíveis. Diversas técnicas são, então, aplicadas para extração de dados e classificação das *features*, como técnicas de extração de segmentos de retas, cantos, etc. Em [Marjovi e Marques 2011], por exemplo, são utilizados os números de segmentos, cantos, manchas (*blobs*) e picos em um histograma para classificação das *features*. Esses dados são obtidos a partir da leitura de um sensor de varredura laser.

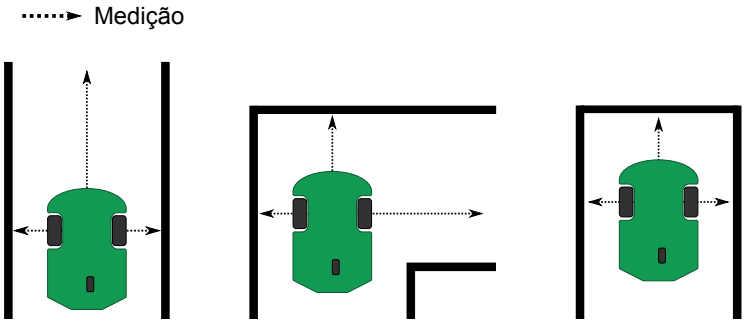
Neste trabalho, a solução adotada para identificação das *features* é bastante simples, não envolvendo algoritmos sofisticados para processamento dos dados sensoriais obtidos. O foco deste trabalho não está no processo de identificação de *features*, então a solução adotada tem apenas o intuito de preencher essa etapa do processo de exploração. As técnicas apresentadas no decorrer do trabalho independem da forma como as *features* são extraídas do ambiente. Métodos mais sofisticados poderão ser adotados.

A identificação de *features* é realizada com apenas três medições pontuais de distância. Essas medições são feitas na frente e nas laterais do robô e cada *feature* está associada a uma combinação diferente dessas três medições. Esse processo de identificação não exige a medição precisa das distâncias; faixas de valores são suficientes para a identificação. A Figura 18 ilustra o processo. Para cada uma das Figuras 18(a), 18(b) e 18(a) uma combinação de faixas de valores é encontrada pelos sensores do robô.

É considerado também que os robôs são capazes de identificar outros robôs, diferenciando-os de obstáculos. Isso evita que o encontro entre dois robôs durante a exploração seja indevidamente identificado como uma parede e, consequentemente, identificado como uma *feature* inexistente na porção do ambiente em questão.

### 4.2.2 Construção do mapa

A construção do mapa ocorre de forma centralizada. Todas as informações coletadas pelos robôs sobre o ambiente são diretamente incorporadas a um mapa global único. Isso é possível pois, no início da exploração, todos os robôs possuem uma mesma referência global (mesmo ponto inicial, por exemplo). Devido a essa referência comum,




(a) Corredor (b) Esquina (c) Dead end

Figura 18: Identificação de *features*

todos os robôs possuem o mesmo sistema de coordenadas.

Alguns trabalhos tratam do problema de exploração sem assumir uma referência global em comum para os robôs [Marjovi e Marques 2011]. Nesses casos, são construídos mapas locais para cada um deles e utilizadas técnicas para fusão dos mesmos. Essa fusão pode ocorrer por similaridade dos mapas locais, busca de um referencial conhecido no ambiente ou até mesmo o encontro intencional dos robôs para alinhamento dos seus sistemas de coordenadas. O problema de fusão de mapas locais, não tratado neste trabalho, é complementar ao de planejamento de trajetória.

Foi assumido que o ambiente estruturado estudado é formado somente por paredes arranjadas de forma paralela e perpendicular. Essas características são usualmente encontradas em construções estruturadas, como escritórios e armazéns, e podem ser facilmente utilizadas pelos robôs para manter informações em relação a sua própria direção. Em [Bando e Yuta 2010] essas características do ambiente são não só assumidas como frequentemente encontradas, como também são utilizadas para correção de erros de estimação na direção dos robôs.

Dados métricos são incorporados aos nós do grafo para auxílio na montagem do mapa. Esses dados auxiliam na determinação de relações de adjacência entre os nós e na resolução de problemas como a determinação da equivalência entre pontos do mapa. Duas fontes de dados métricos são utilizadas: a medição do comprimento dos corredores e a orientação das *features*.

Os corredores tem seu comprimento medido pelos robôs, neste trabalho, utilizando odometria. Esta apresenta erros incrementais, e

sua utilização em longos períodos de tempo faz com que as incertezas sejam muito elevadas, impossibilitando uma utilização confiável. A realização da medição somente de corredores, entretanto, não permite que essa incerteza seja muito elevada, pois os intervalos de medição são curtos e ao fim de cada corredor a odometria é reiniciada. A odometria, entretanto, é considerado como um caso conservador. Diversas outras técnicas de medição de deslocamento dos robôs poderiam ser utilizadas, como a utilização de unidades de medição inercial, por exemplo.

Devido às características do ambiente assumido, pode-se considerar que o robô transiciona entre quatro possíveis orientações discretas. Com a informação da orientação do robô é possível determinar se os corredores estão orientados, num plano global, de forma vertical ou horizontal. Conhecida a posição inicial do robô, o comprimento dos corredores e suas orientações, já é possível realizar a montagem de um esqueleto do mapa.

Entre os corredores estão localizadas as demais *features*: os nós do grafo onde os corredores são os arcos. A esses nós são associadas outras informações métricas, como suas coordenadas e orientação. As coordenadas são obtidas a partir de um cálculo considerando o comprimento e direção dos corredores. Já a orientação é obtida a partir da orientação discreta do robô ao acessar a *feature*.

A posição do nó é armazenada para os nós associados a qualquer tipo de *feature*. Nós associados a algumas *features*, entretanto, não necessitam da sua orientação armazenada. Cruzamentos podem ser acessados a partir de qualquer uma das quatro orientações discretas do robô e são sempre identificados simplesmente como "cruzamento". Dessa forma, não necessitam da informação de orientação global. Já becos-sem-saída podem ser acessados por apenas uma das orientações discretas, sendo armazenada como sua orientação global a orientação do robô quando o acessou. Se um robô orientado para o norte, por exemplo, acessa um beco-sem-saída este é salvo com um "beco-sem-saída orientado para o norte".

Com esquinas e junções o processo é um pouco mais complicado. Uma esquina, por exemplo, pode ser identificada como "esquina à direita" ou "esquina à esquerda". Entretanto, essas estruturas são as mesmas, e o que define sua orientação global é a orientação do robô quando ele a acessa. Nas Figuras 19(a), 19(b), 19(c) e 19(d) são mostradas as quatro diferentes orientações da *feature* esquina.

Assim, por exemplo, se um robô orientado para o norte identifica uma "esquina à direita" temos uma esquina com a orientação mostrada na Figura 19(a) acessada pelo ponto A, que é o mesmo caso em que um

robô orientado para oeste encontra uma "esquina à esquerda" (Figura 19(a) acessada por B).

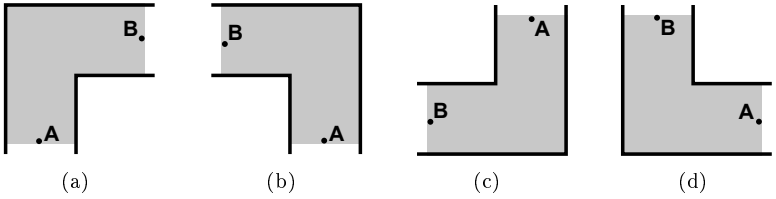



Figura 19: Diferentes orientações da *feature* esquina

Junções podem ser identificadas como "junção à direita", "junção à esquerda" e "junção em T". Nas Figuras 20(a), 20(b), 20(c) e 20(d) são apresentadas as quatro diferentes orientações da *feature* junção. A junção da Figura 20(a), por exemplo, é identificada como uma "junção à esquerda" quando acessada por A, como uma "junção em T" quando acessada por B e "junção à direita" quando acessada por C.

De forma resumida temos que para os corredores é armazenado o comprimento e para as demais *features* as posições (coordenada x, y). Já a orientação ocorre da seguinte forma:

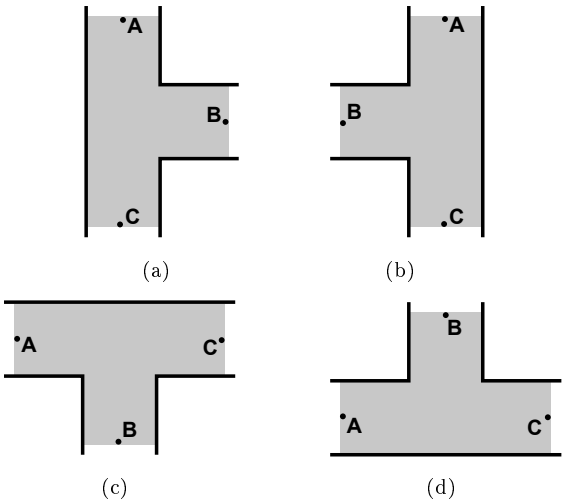



Figura 20: Diferentes orientações da *feature* junção

**Cruzamento:** Uma possível orientação global; sempre interpretado pelos robôs como "cruzamento";

**Beco-sem-saída:** Quatro possíveis orientações; sempre interpretado pelos robôs como "beco-sem-saída";

**Esquina:** Quatro possíveis orientações; pode ser interpretado como "esquina à direita" ou "esquina à esquerda";

**Junção:** Quatro possíveis orientações; pode ser interpretado como "junção à esquerda", "junção à direita" e "junção em T";

**Corredor:** Duas possíveis orientações (horizontal e vertical); sempre interpretado pelos robôs como "corredor";

Cada nó possui um número fixo de nós adjacentes dependendo da *feature* a qual está associado. Duas *features* conectadas a extremidades opostas de um mesmo corredor são representadas no grafo por dois nós conectados por um arco. Uma esquina possui dois nós adjacentes, um cruzamento quatro e assim por diante. Os nós adjacentes estão sempre localizados em uma das quatro possíveis orientações discretas. O nó associado a *feature* junção da Figura 20(a), por exemplo, possui três nós adjacentes: um na posição norte, um na posição leste e um na posição sul.

Quando uma nova *feature* é descoberta e um nó associado a ela é adicionado no grafo, o número e posição dos nós adjacentes é automaticamente conhecido. O tipo desses nós adjacentes é então marcado como "desconhecido" e permanece dessa forma até que um robô visite a região correspondente, identificando o tipo da sua *feature*.

O método proposto para identificação das *features* e construção do mapa não considera a existência de incertezas associadas à identificação das *features*. Uma vez identificada uma *feature* e associada a um nó do grafo ela é assumida como correta e completamente conhecida e torna-se desnecessário conferir novamente o seu "tipo".

### 4.2.3 Determinação de equivalência

Um desafio recorrente na construção de mapas de ambientes desconhecidos é a determinação de equivalência entre pontos desse mapa. Esse desafio, também conhecido como "fechamento de *loop*", é mais evidente em abordagens topológicas por, tipicamente, não utilizarem dados da posição das regiões do ambiente na representação do mapa.

Durante a exploração de um ambiente desconhecido, o conjunto de sensores de um robô pode identificar as mesmas características ambientais em momentos diferentes. Essas características podem estar relacionadas com regiões diferentes do mapa, porém, podem também estar associadas a uma mesma região atingida pelo robô através de caminhos diferentes. Dados da posição dessas regiões ajudam no processo de eliminação de ambiguidade, porém, devido aos erros de medição, mesmo com a utilização desses dados o problema ainda está presente. Na Figura 21 é exemplificado o problema: o robô parte da esquina B e executa o trajeto indicado, chegando na esquina A. Devido aos erros de medição é difícil concluir se o robô retornou ao ponto inicial (A e B são iguais) ou se ele atingiu uma nova esquina (A e B são diferentes).

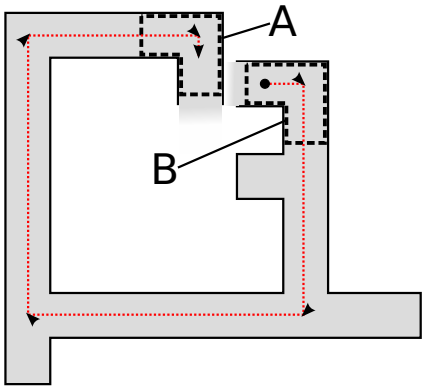



Figura 21: Problema do fechamento de *loop*

No método proposto neste trabalho, o problema da determinação de equivalência é resolvido através da comparação do nó descoberto com uma série de nós candidatos. Os nós que possuem um nó adjacente ainda desconhecido são mantidos em uma lista de nós que podem ser atingidos durante a exploração através de um dos nós desconhecidos. Para cada nova *feature* descoberta é adicionado um nó no grafo e, então, é realizado um processo de comparação desse nó com os elementos da lista, verificando a equivalência entre eles. Para que a equivalência seja determinada, o processo ocorre em três etapas:

1. O tipo do nó associado a *feature* descoberta é comparado com o tipo dos nós da lista. Para que essa etapa seja verdadeira é comparado também se a orientação global dos nós é compatível;

2. Para que os nós sejam equivalentes, a *feature* descoberta deve ter sido acessada pela mesma direção na qual os nós da lista tenham um nó adjacente desconhecido associado;

3. Os nós devem ter coordenadas globais próximas, ou seja, os nós não podem ter distância maior que um limite especificado compatível com o erro dos sensores.

O Algoritmo 1 ilustra o processo de teste de equivalência quando descoberta uma nova *feature* no ambiente. Um nó associado a essa *feature* é criado e então comparado com uma lista de possíveis candidatos. Caso o algoritmo determine que não há equivalência do novo nó com os já existentes, o novo nó é mantido e a exploração prossegue. Se verificada a equivalência entre dois nós, o novo nó é eliminado, são determinadas as novas relações de adjacência e são corrigidas as coordenadas dos nós pertencentes ao *loop*.

### 4.2.4 Exemplo ilustrativo da construção de mapa

Na Figura 22 é apresentado um exemplo do processo de mapeamento. O processo inicia com o robô localizado em uma *feature* do tipo beco-sem-saída e com a criação de um grafo com o nó 0 e coordenada $(0, 0)$ associado ao ponto de partida do robô, conforme apresentado nas Figuras 22(a) e 22(b). Como o nó está associado a uma *feature* do tipo beco-sem-saída e assumindo conhecida a direção inicial do robô, é possível concluir que existe apenas um nó adjacente ao nó atual e também a direção que ele está localizado.

Nas Figuras 22(c) e 22(d) o processo de mapeamento continua percorrendo-se o corredor, descobrindo a *feature* cruzamento e adicionando o nó correspondente no grafo (nó 1). O comprimento do corredor é medido, possibilitando calcular a posição relativa do nó 1 em relação ao nó 0. As coordenadas do nó 1 são incorporadas ao grafo, assim como a posição dos três nós adjacentes desconhecidos. Como o nó 1 possui nós adjacentes ainda desconhecidos, ele é mantido em uma lista de nós que podem ser atingidos a partir de um dos nós desconhecidos.

Nas Figuras 22(e) e 22(f), com a descoberta de novas *features* no ambiente, os nós 2, 3, 4 e 5, correspondentes a essas *features*, são adicionados ao grafo. Cada nó adicionado ao grafo é comparado com a lista de nós com ao menos uma adjacência desconhecida e a equivalência dos nós é testada. Para os nós de 2, 3 e 4 o teste é encerrado logo na primeira condição, pois enquanto na lista é mantido um nó do tipo

**Algoritmo 1:** Teste de equivalência de nós

```
   input  : ListaNosCandidatos, NovoNo
   output: NoEquivalente
 1 tipo ← Tipo(NovoNo)
 2 direcaoAcesso ← Direcao(NovoNo)
 3 orientacao ← OrientacaoGlobal(NovoNo)
 4 posicao ← Posicao(NovoNo)
 5 α ← Tolerancia de distancia
 6 menorDistancia ← ∞
 7 for NoCandidato : ListaNosCandidatos do
 8 |   if Tipo(NoCandidato)≠ tipo then
 9 |   |   Remove NoCandidato de ListaNosCandidatos
10 |   end
11 |   if orientacao ≠ OrientacaoGlobal(NoCandidato) then
12 |   |   Remove NoCandidato de ListaNosCandidatos
13 |   end
14 |   if NoCandidato(direcaoAcesso + 180^o) ≠ Desconhecido
   |   then
15 |   |   Remove NoCandidato de ListaNosCandidatos
16 |   end
17 end
18 for NoCandidato : ListaNosCandidatos do
19 |   if posicao − Posicao(NoCandidato) < α then
20 |   |   if
   |   |   posicao−Posicao(NoCandidato) < menorDistancia
   |   |   then
21 |   |   |   menorDistancia =
   |   |   |   posicao − Posicao(NoCandidato)
22 |   |   |   NoEquivalente = NoCandidato
23 |   |   end
24 |   end
25 end
26 if NoEquivalente = null then
27 |   return null
28 |   Imprime: "nenhum nó equivalente encontrado"
29 end
30 else
31 |   return NoEquivalente
32 |   Imprime: "nó equivalente = NoEscolhido"
33 end
```

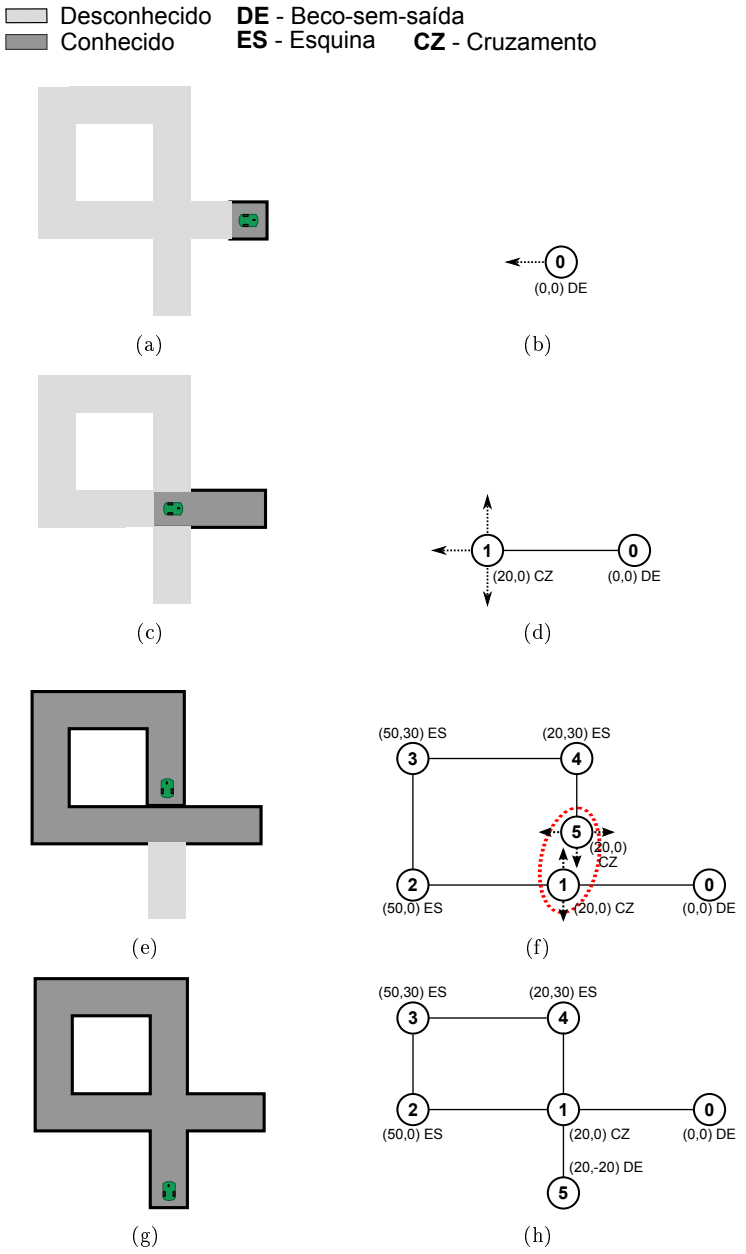



Figura 22: Exemplo do processo de mapeamento

cruzamento (nó 1) os nós descobertos são do tipo esquina.

Com a identificação de outra *feature* do tipo cruzamento o nó 5 é adicionado ao grafo e um novo teste de equivalência é executado. O nó 1, mantido na lista de nós com adjacência desconhecida, possui mesmo tipo que o nó sendo testado (nó 5). Como o tipo "cruzamento" não possui informação de orientação global o teste passa pela primeira etapa. Como o nó 5 foi acessado pela sua adjacência norte e o nó 1 possui sua adjacência norte desconhecida, o teste também passa pela segunda etapa. Por último, é comparada a distância entre os nós a partir das coordenadas associadas a eles. De acordo com a Figura 22(f), os nós possuem coordenadas idênticas, validando a terceira etapa do teste. Caso os nós possuíssem coordenadas diferentes, a diferença da posição deles seria comparada a um limite estabelecido arbitrariamente. Se a diferença for menor que o valor limite a terceira etapa do teste também é validada. Como o resultado do teste de equivalência foi positivo, o nó 5 é deletado e a relação de adjacência entre os nós 1 e 4 é criada.

Após a determinação de equivalência entre os nós 1 e 4 o robô se dirige para a posição ainda não explorada do cruzamento associado ao nó 2. Um novo nó identificado como 5 é adicionado ao grafo, associado ao beco-sem-saída adjacente ao cruzamento.

Nas Figuras 22(g) e 22(h) é mostrado o resultado final do processo de mapeamento. Na Figura 22(g) é apresentado o ambiente completamente explorado e na Figura 22(h) é mostrado o grafo correspondente construído, no qual é também exibido o tipo e posição dos nós.

## 4.3 ALOCAÇÃO DE TAREFAS

O alocador de tarefas é responsável por determinar qual ponto do mapa deverá ser explorado por cada robô, dada a distribuição dos robôs no mapa parcial já construído. Esses pontos a serem explorados são os objetivos locais e a realização de todos os objetivos locais resulta também na conclusão do objetivo global do sistema, que é a exploração total do ambiente.

As técnicas empregadas na alocação de tarefas tem impacto direto no tempo total de exploração do ambiente. Se os robôs forem enviados primeiramente para as regiões mais distantes do ambiente, por exemplo, em um segundo momento eles terão que retornar por caminhos já explorados, executando trabalho redundante. Assim, para que a exploração do ambiente desconhecido ocorra de forma eficiente,

diversos fatores devem ser observados durante o desenvolvimento do alocador de tarefas.

Para a definição do significado de tarefa neste trabalho é importante abordar o conceito de *fronteiras*. ***Fronteiras* são os limites entre regiões exploradas e ainda não exploradas do ambiente** [Yamauchi 1997], ou seja, as saídas de uma *feature* as quais não foram atravessadas por um robô. A Figura 23(a) identifica a presença de fronteiras em um ambiente parcialmente explorado.

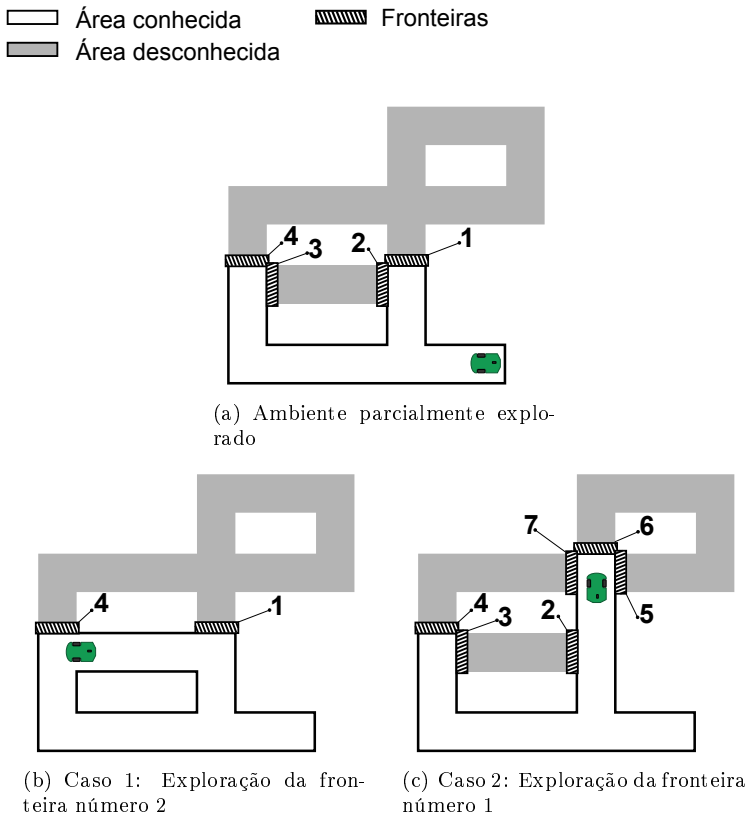
**As tarefas são definidas como o ato de deslocar-se até uma fronteira e explorá-la, afim de obter informações do ambiente para compor o mapa.** A execução de todas as tarefas faz com que o objetivo global do sistema seja cumprido, que é a exploração total do ambiente desconhecido.

A exploração de uma fronteira faz com que ela seja projetada para frente à medida que o robô avança sobre a área desconhecida. Durante esse processo, novas fronteiras podem ser descobertas e, tendo o alocador que designar apenas uma delas para um robô, as demais têm que ser revisitas em um momento posterior. Sempre que descobertas novas fronteiras o nó do grafo associado a *feature* em questão é adicionado a uma lista de tarefas, utilizada no processo de alocação.

A Figura 23 ilustra o processo de exploração de fronteiras. Na Figura 23(a) é mostrado um ambiente parcialmente explorado, com identificação de quatro fronteiras. As fronteiras 1 e 2 estão localizadas em uma *feature* do tipo junção e as fronteiras 3 e 4 a outra *feature*, também do tipo junção. Os nós do grafo associados a essas *features* são mantidos na lista de tarefas.

Na Figura 23(b) a fronteira identificada como 2 é alocada para o robô. Sua exploração resulta no fechamento de um *loop*, e na também exploração da fronteira 3. Como ainda existem fronteiras associadas às junções mencionadas, os nós associados a elas continuam na lista de tarefas.

Já na Figura 23(c), se o robô for designado para explorar a fronteira 1, ao fim da exploração será descoberta uma *feature* cruzamento. Em função disso, três novas fronteiras são descobertas, identificadas na figura como 5, 6 e 7. O nó associado a esse cruzamento é, então, adicionado à lista de tarefas.




(a) Ambiente parcialmente explorado

(b) Caso 1: Exploração da fronteira número 2

(c) Caso 2: Exploração da fronteira número 1

Figura 23: Exploração de fronteiras

### 4.3.1 Processo de alocação

A alocação de tarefas é realizada baseando-se em uma função custo que determina valores para cada uma das fronteiras. Esses valores são determinados por heurísticas, apresentadas na Seção 4.3.2 e representam qual a tarefa mais vantajosa a ser executada por um robô, baseando-se em fatores como a distância até a fronteira e a quantidade de informações que a exploração da fronteira fornecerá para a construção do mapa.

A tarefa do alocador consiste em encontrar a distribuição de tarefas entre os robôs que apresente o menor valor da função custo. Neste trabalho a alocação de tarefas é classificada em dois grupos: a

alocação para um único robô e a alocação para todos os robôs.

Diversos trabalhos utilizam o conceito de função custo para tratar do problema de exploração de ambientes desconhecidos, como [Burgard et al. 2005]. Este trabalho, entretanto, propõe heurísticas para avaliar o custo de cada tarefa, calculadas de forma particular, e o método proposto utiliza um procedimento de alocação que busca diminuir a quantidade de vezes que a alocação de tarefas tem que ser realizada para todos os robôs.

O tempo que cada robô leva para atingir e explorar uma fronteira associada a uma tarefa é, em geral, diferente dos demais. Para que a alocação seja realizada de forma eficiente ela deve ser realizada avaliando quais as possibilidades para todos os robôs e optando pela configuração que irá ser realizada de forma potencialmente mais rápida. Isso pode ser feito aguardando que todos os robôs terminem a realização de uma tarefa para, então, realizar a próxima alocação.

Esse processo, entretanto, faz com que os primeiros robôs a terminarem a execução de tarefa fiquem ociosos até que o último tenha também terminado. Para evitar esse período de ociosidade, sempre que possível uma nova tarefa é alocada aos robôs à medida que eles executam a tarefa anterior. Essa é a alocação para um único robô.

Em geral, a realização de uma tarefa revela a existência de fronteiras desconhecidas, e devido ao fato do robô já estar localizado ao lado dessas fronteiras o processo de alocação é bastante simples. Nestes casos, o alocador de tarefas tem apenas que realizar uma decisão da direção que o robô deve seguir.

Entretanto, existem situações em que a realização de uma tarefa não finaliza no descobrimento de novas fronteiras. Essas situações ocorrem quando a realização de uma tarefa finaliza nas seguintes formas:

- Descobrimento de um *dead end*;
- Fechamento de um *loop*;
- Encontro entre dois robôs ao realizar a exploração de um corredor partindo de extremidades opostas.

Nestes casos, as tarefas a serem alocadas estão distantes do robô e uma análise de qual delas possui o menor custo e qual robô deve realizá-la é mais complexa. O alocador aguarda que todos os robôs finalizem a navegação pela *feature* corrente para então avaliar os custos para realização das próximas tarefas. Isso evita que a alocação ocorra enquanto o robô não possui uma posição definida no grafo correspon-

dente ao mapa do ambiente. Durante a navegação por um corredor, por exemplo, o robô estaria em transição entre dois nós do grafo.

O diagrama da Figura 24 mostra o processo de alocação de tarefas, destacando a decisão entre a necessidade de realizar uma alocação para todos os robôs e uma simples decisão de direção para um único robô. Quando existe a necessidade de realizar a alocação de tarefas para todos os robôs a alocação só é realizada assim que todos os robôs finalizam a navegação pela feature corrente.

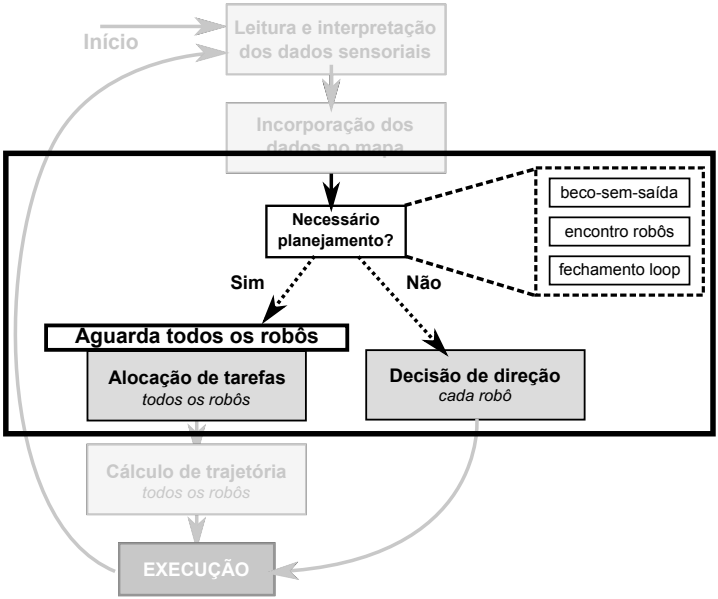



Figura 24: Fluxograma do processo de exploração proposto

Uma vez decidido que será realizada a alocação para todos os robôs devido ao acontecimento de uma das situações listadas na Figura 24, é iniciado o processo apresentado no Algoritmo 2. Esse processo inicia aguardando que todos os robôs finalizem a exploração da *feature* corrente.

Com todos os robôs disponíveis, uma ordem aleatória de prioridade entre eles é estabelecida para a realização da alocação. Para cada robô é calculado o custo para execução de cada uma das tarefas disponíveis e, então, a tarefa de menor custo é selecionada. O custo total da alocação de tarefas para todos os robôs é calculado somando-se o custo

de cada uma das tarefas selecionadas e uma nova ordem de prioridade entre os robôs é proposta. Se a nova ordem de prioridade proposta apresentar uma solução de menor custo essa ordem é mantida e, caso contrário, ela é descartada. Esse processo se repete por um número arbitrário de vezes, definido pela variável $\delta$.

A alocação de uma tarefa para um robô influencia na alocação das demais, pois os custos para realização das tarefas são alterados de acordo com as heurísticas utilizadas apresentadas na Seção 4.3.2. Dessa forma, o estabelecimento de uma ordem para alocação das tarefas tem forte influencia no resultado final da alocação. Como o processo de determinação de uma ordem de prioridade entre os robôs é realizado, neste trabalho, de forma aleatória, o número de vezes que esse processo é repetido tem também influencia no resultado final da alocação. Neste trabalho o número de repetições é definido de forma arbitrária.

Na Figura 25 é apresentado um exemplo ilustrativo do processo de alocação de tarefas. Na Figura 25(a) as fronteiras 1 e 2 são alocadas para os robôs indicados. Nas Figuras subsequentes, 25(b) e 25(c), o robô $A$ finaliza a exploração da fronteira 1, resultando no descobrimento de um beco-sem-saída.

Como uma tarefa foi concluída é avaliada a necessidade de um novo planejamento. O descobrimento de um beco-sem-saída indica a necessidade de uma alocação de tarefas para todos os robôs. O processo de alocação, então, aguarda que o robô $B$ finalize a exploração da *feature* corrente.

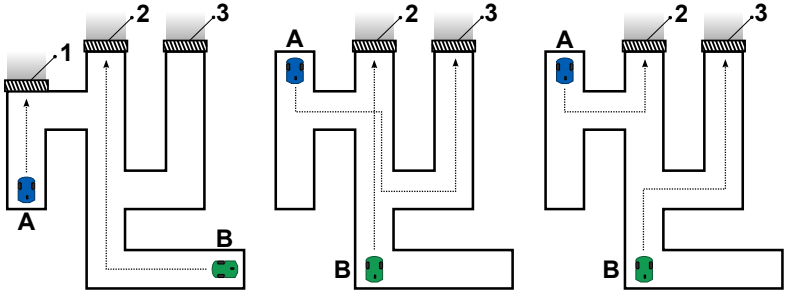



Figura 25: Processo de alocação de tarefas

As Figuras 25(b) e 25(c) mostram o resultado da alocação para duas possíveis ordens de prioridade entre os robôs. Nesse exemplo, é considerado como custo da tarefa apenas a distância entre o robô e a

**Algoritmo 2:** Alocação de tarefas

```
   input : ListaRobos, ListaTarefas
 1 melhorCusto ← ∞
 2 custoTotal = 0
 3 δ = número arbitrário de iterações do alocador
 4 ListaCustos = empty
 5 while nRobosReady ≠ size(ListaRobos) do
 6 |  wait
 7 end
 8 for i = 0; i < δ do
 9 |  custoTotal = 0
10 |  shuffle(ListaRobos)
11 |  for j = 0; j < size(ListaRobos) do
12 |  |  r = elemento j da ListaRobos
13 |  |  j = 0
14 |  |  for t : ListaTarefas do
15 |  |  |  ListaCustos(j) = custo para o robô r realizar a
   |  |  |  tarefa t
16 |  |  |  j = j + 1
17 |  |  end
18 |  |  melhorTarefa = Tarefa de ListaCustos de menor
   |  |  custo
19 |  |  custo = custo de melhorTarefa
20 |  |  r ← melhorTarefa
21 |  |  custoTotal = custoTotal + custo
22 |  end
23 |  if custoTotal < melhorCusto then
24 |  |  melhorCusto = custoTotal
25 |  |  melhorSequencia = ListaRobos
26 |  end
27 end
```

fronteira associada à tarefa em questão.

Na Figura 25(b) o robô $B$ é alocado primeiro, e a tarefa designada a ele é a associada a fronteira número 2 por ser a de menor custo (menor distância, neste exemplo). O robô $A$ é, consequentemente, alocado à tarefa associada à fronteira número 3.

Na Figura 25(c) o robô $A$ é alocado primeiramente, sendo selecionada a tarefa de menor custo (fronteira número 2). O robô $B$ é, então, alocado à fronteira número 3.

Nas Figuras 25(b) e 25(c) são também identificados os trajetos a serem percorridos pelos robôs, uma vez alocadas as respectivas tarefas. Como nesse exemplo foi apenas considerado como custo a distância, o custo total de cada solução é a soma da distância a ser percorrida por cada robô. Adotando ordens de prioridades diferentes o custo total da solução pode ser diferente, como pode ser facilmente visualizado nas Figuras 25(b) e 25(c).

### 4.3.2 Heurísticas

A alocação de tarefas é um problema que possui elevada complexidade teórica. Devido ao fato do ambiente além das fronteiras ser desconhecido, a distribuição de tarefas para os robôs de forma a garantir o menor tempo de exploração possível depende de inúmeros fatores, fazendo que o grande número de possibilidades leve a uma explosão combinatória. Dessa forma, o problema não pode ser resolvido com métodos de programação convencionais, principalmente os de natureza puramente numérica.

Nesse sentido, uma alternativa para a resolução do problema é a utilização de heurísticas. Um procedimento heurístico é um método baseado em experiência e conhecimento do problema, que não segue um percurso claro, mas se baseia na intuição e nas circunstâncias para resolver ou simplificar o problema de forma empírica. Heurísticas, no entanto, não podem ser verificadas formalmente, fazendo com que não exista uma prova matemática apontando que ela possui propriedades específicas.

As heurísticas propostas neste trabalho para a alocação de tarefas se concentram na diminuição da passagem dos robôs por áreas já exploradas, evitando trabalho redundante, e também da interação entre robôs, que causa movimentações desnecessárias. Essas heurísticas são aplicadas ao problema de alocação de tarefas através de uma função custo, seguida por uma busca realizada para encontrar a distribuição

de tarefas entre os robôs que apresente o menor custo. Como cada uma das heurísticas busca diminuir o tempo de exploração trabalhando com aspectos diferentes do sistema, estima-se que a combinação delas forneça resultados melhores que a utilização das heurísticas isoladamente. Esse ponto será investigado no capítulo 5.

Cada uma das heurísticas fornece um valor de custo para cada fronteira, representando a vantagem ou dificuldade de atingi-la. Cada uma delas possui uma maneira particular de ser calculada, dependendo da sua natureza. Os cálculos dos custos de cada heurística serão apresentados nas seções subsequentes. O custo para atingir a fronteira em questão é, então, composto pelo custo de cada uma das heurísticas.

As heurísticas tratadas nesse trabalho são: menor distância ($A$), distribuição dos robôs ($B$) e fechamento de *loop* ($C$). A Equação 4.1 mostra a composição do custo para atingir uma fronteira, na qual $\alpha$, $\beta$ e $\gamma$ são constantes para que cada heurística tenha participação igual na composição do custo. Os valores dessas constantes foram atribuídos através da observação dos resultados experimentais e são discutidos na Seção 5.2.1.

$$custo = \alpha A + \beta B + \gamma C \tag{4.1}$$

#### 4.3.2.1 Menor distância

A ideia mais intuitiva de distribuição de tarefas é a de designar a cada robô a exploração da fronteira mais próxima a ele. Apesar de bastante simples, na prática essa alocação faz com que as fronteiras mais próximas sejam exploradas primeiro, evitando que os robôs tenham que atravessar caminhos já explorados para retornar a essas fronteiras em outro momento. Essa alocação também evita que a distribuição de um grupo de tarefas a um grupo de robôs seja feita de forma que os robôs tenham que percorrer um caminho médio mais longo que o necessário.

Quando é calculado um caminho entre dois nós do grafo, a solução obtida é uma lista sequencial de nós a serem percorridos. O cálculo do custo da heurística de menor distância é mostrado na Equação 4.2, na qual são somadas as distâncias entre os nós a serem percorridos, partindo da posição inicial do robô até a posição do nó correspondente a fronteira em questão. A Equação 4.2 aplica-se somente a ambientes estruturados com as características apontadas neste trabalho, ou seja, formados apenas por paredes paralelas e perpendiculares umas às outras.

Nessa equação as variáveis $x$ e $y$ se referem as coordenadas de posição dos nós e o índice $i$ ao elemento da lista sequencial de nós a serem percorridos. A expressão $|x_i - x_{i+1}|$ representa, então, a diferença em "$x$" entre dois nós subsequentes; e a expressão $|y_i - y_{i+1}|$ a diferença em "$y$".

$$\sum_{i=posInicial}^{posFinal-1} |x_i - x_{i+1}| + |y_i - y_{i+1}| \tag{4.2}$$

Na Figura 26 são mostrados os robôs $A$, $B$ e $C$ em posições iniciais diferentes dentro de um mesmo mapa. São mostrados também os custos para atingir cada uma das fronteiras a partir das respectivas posições iniciais. Os custos indicados são proporcionais à distância a ser percorrida pelos robôs através do caminho mais curto para atingir a fronteira em questão. A partir dessa informação, a alocação é feita buscando minimizar o custo total para alcançar essas fronteiras.

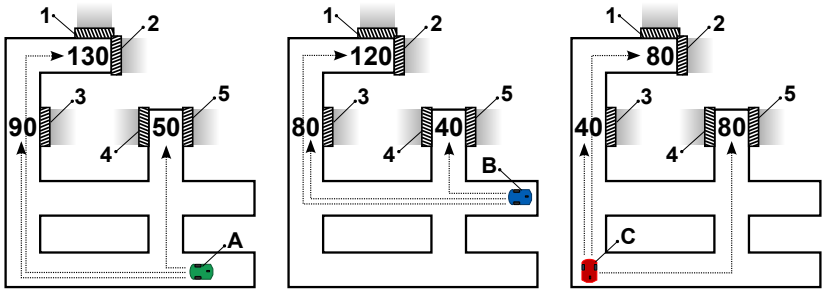



Figura 26: Custos associados à distância para atingir as fronteiras

Um exemplo da alocação de tarefas utilizando a heurística de menor distância, considerando os robôs $A$, $B$ e $C$ da Figura 26 em no mesmo ambiente, ocorreria da seguinte forma:

- Robô $A$ é alocado à fronteira 4, com custo igual a 50;
- Robô $B$ é alocado à fronteira 5, com custo igual a 40;
- Robô $C$ é alocado à fronteira 3, com custo igual a 40;

O custo total dessa alocação é de 130 $(50 + 40 + 40)$. Em outra situação, sem utilizar a heurística de menor distância, poderíamos ter a seguinte situação:

- Robô $A$ é alocado à fronteira 1, com custo igual a 130;
- Robô $B$ é alocado à fronteira 2, com custo igual a 120;
- Robô $C$ é alocado à fronteira 5, com custo igual a 80;

Totalizando em um custo de 330 $(130 + 120 + 80)$, cerca de 2.5 vezes mais elevado.

#### 4.3.2.2 Distribuição dos robôs

A distribuição dos robôs pelo mapa, mantendo-os afastados uns dos outros, também é uma característica desejada durante a exploração de ambientes desconhecidos e pode ser alcançada através da utilização de uma heurística. A proximidade entre os robôs aumenta o grau de interferência entre os mesmos, provocando a inclusão de desvios e de paradas de espera em suas trajetórias. Além disso, a realização da exploração de uma mesma região do ambiente por múltiplos robôs propicia um aumento da passagem por caminhos já explorados. Todos esses fatores prejudicam a eficiência do sistema de exploração provocando um aumento no tempo total necessário para realização da atividade.

Como a alocação de tarefas é feita sequencialmente para cada robô, a primeira alocação não utiliza dados da heurística de distribuição para escolha da tarefa. Porém, uma vez realizada a primeira alocação, a fronteira selecionada propaga um custo adicional às demais fronteiras. O valor desse custo é inversamente proporcional à distância da fronteira alocada em relação às demais. A distância utilizada para o cálculo dos custos, neste caso, é a distância Euclidiana. Isso é possível devido ao fato de valores das coordenadas estarem associados aos nós do grafo.

A Equação 4.3 mostra o cálculo propagado por uma fronteira $i$ em uma fronteira $j$, após ser alocada a um robô.

$$custo = \frac{1}{\sqrt{|x_i - x_j|^2 + |y_i - y_j|^2}} \tag{4.3}$$

A justificativa para utilização da distância Euclidiana reside no fato de que o ambiente é desconhecido e que as reais distâncias entre as fronteiras não são conhecidas. A utilização da distância considerando somente o ambiente conhecido fornece uma estimativa fraca da real distância entre as fronteiras. As fronteiras 2 e 4 da Figura 27 exemplificam essa situação. Apesar da distância entre elas através do ambiente conhecido ser grande, elas estão de fato bem próximas uma da

outra. Robôs explorando essas fronteiras simultaneamente tem grande chance de se encontrarem, uma vez que podem haver conexões ainda desconhecidas entre essas regiões.

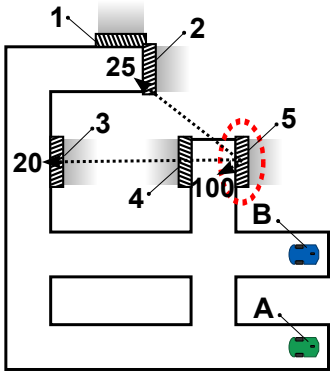



Figura 27: Custos propagados pela alocação de uma fronteira

Na Figura 27 são mostrados os custos propagados às fronteiras em função da alocação da fronteira 5 ao robô $B$, por exemplo. Na alocação para o robô $A$ os custos propagados por essa fronteira são considerados. A fronteira 4 se destaca nesse contexto, pois possui o maior custo propagado (igual a 100) devido a sua proximidade com a fronteira 5.

Além da distância entre as fronteiras, a heurística de distribuição dos robôs considera também a direção da fronteira a ser explorada. Ainda na Figura 27, as fronteiras 1 e 2 possuem a mesma distância em relação à fronteira alocada (fronteira 5). Apesar disso, a exploração da fronteira 1 irá afastar o robô da fronteira alocada, enquanto a exploração da fronteira 2 irá aproximá-lo. Assim, a fronteira 1 possui menor custo que a fronteira 2, pois se adota na heurística que a busca de áreas mais afastadas é prioritária.

#### 4.3.2.3 Fechamento de loop

Ainda com a intenção de evitar a passagem dos robôs por regiões já exploradas do ambiente, reduzindo o trabalho redundante, é proposta a heurística de fechamento de *loop*. Essa heurística busca dar prioridade à alocação de fronteiras cuja exploração da *feature* associada seja, potencialmente, finalizada em outra fronteira conhecida.

Essa situação pode ser exemplificada através da Figura 27, na qual as fronteiras 3 e 4 estão alinhadas e possivelmente estão conectadas a um mesmo corredor. Quanto maior a região explorada no entorno dessas fronteiras, maior é a distância a ser percorrida pelos robôs por regiões já exploradas para retornar a elas, por isso é desejável que o fechamento desses *loops* ocorra assim que as fronteiras em questão sejam descobertas. Além disso, o fechamento de *loops* fornece alternativas de caminhos para os demais robôs que tem que se deslocar pelo ambiente.

O cálculo do custo associado a essa heurística é iniciado verificando se a exploração da fronteira sendo analisada irá potencialmente ser finalizada no fechamento de um *loop*. Esse processo é realizado avaliando se existem outras fronteiras com direção de exploração oposta à fronteira em questão, se essas fronteiras possuem coordenada $x$ ou $y$ similares e se as fronteiras estão fisicamente próximas. Em caso positivo é tomada a fronteira mais próxima à fronteira analisada que atenda aos requisitos, e a distância entre as mesmas é utilizada como o custo da heurística de fechamento de *loop*.

O Algoritmo 3 ilustra esse processo. Quando não é identificado nenhuma fronteira que potencialmente resulte no fechamento de um *loop* o algoritmo retorna um valor padrão. Caso seja identificado, esse valor é substituído pela distância entre as fronteiras, menor que o valor padrão adotado.

Utilizando a Figura 27 como exemplo, o custo da heurística de fechamento de *loop* seria o valor padrão para as fronteiras 1, 2 e 5. Para as fronteiras 3 e 4 o custo seria o valor da distância entre as fronteiras, um valor inferior ao valor padrão.

## 4.4 CÁLCULO DE TRAJETÓRIA

Depois de alocadas as tarefas aos robôs, é necessário calcular o caminho que cada robô deve percorrer para realizar a tarefa designada. Esse caminho corresponde a um conjunto de ações discretas calculadas sobre o grafo correspondente ao mapa do ambiente, devendo levar os robôs às posições desejadas de forma coordenada, sem colisões ou bloqueios.

Os métodos existentes para o planejamento de trajetória de múltiplos robôs compartilhando um mesmo espaço de trabalho podem ser classificados, de uma forma geral, como *centralizados* e *desacoplados* [Bennewitz 2004].

Na abordagem centralizada as possíveis configurações de cada

**Algoritmo 3:** Heurística de fechamento de *loop*

```
   input  : NoAnalisado, ListaNosCandidatos
   output: distancia
 1 α ← Tolerância de distância
 2 distanciaPadrao ← Valor alto
 3 distancia ← Valor alto
 4 for direcao : (0^o, 90^o, 180^o, 270^o) do
 5 |   if NoAnalisado(direcao) ≠ desconhecido then
 6 |   |   próxima iteração
 7 |   end
 8 |   for NoTeste : ListaNosCandidatos do
 9 |   |   if NoTeste(direcao + 180^o) ≠ desconhecido then
10 |   |   |   próxima iteração
11 |   |   end
12 |   |   if NoAnalisado.y = NoTeste.y ± α then
13 |   |   |   if (direcao = 0^o and
   |   |   |   NoTeste.x > NoAnalisado.x) or
   |   |   |   (direcao = 180^o and
   |   |   |   NoTeste.x < NoAnalisado.x) then
14 |   |   |   |   if |NoTeste.x − NoAnalisado.x| < distancia
   |   |   |   |   then
15 |   |   |   |   |   distancia = |NoTeste.x − NoAnalisado.x|
16 |   |   |   |   end
17 |   |   |   end
18 |   |   end
19 |   |   if NoAnalisado.x = NoTeste.x ± α then
20 |   |   |   if (direcao = 90^o and
   |   |   |   NoTeste.y > NoAnalisado.y) or
   |   |   |   (direcao = 270^o and
   |   |   |   NoTeste.y < NoAnalisado.y) then
21 |   |   |   |   if |NoTeste.y − NoAnalisado.y| < distancia
   |   |   |   |   then
22 |   |   |   |   |   distancia = |NoTeste.y − NoAnalisado.y|
23 |   |   |   |   end
24 |   |   |   end
25 |   |   end
26 |   end
27 end
28 if distancia > distanciaPadrao then
29 |   distancia = distanciaPadrao
30 end
31 return distancia
```

robô (posições discretas) são combinadas de forma a se obter um único espaço de estados para todos os robôs e, então, uma busca é realizada nesse espaço para se encontrar uma solução para todo o sistema. Já na abordagem desacoplada o sistema primeiramente calcula um caminho para cada robô omitindo a existência dos demais e, numa etapa seguinte, aplica heurísticas para resolver possíveis conflitos entre os caminhos de diferentes robôs.

Existem dois importantes critérios para avaliar os métodos de planejamento de trajetória [Bennewitz 2004]:

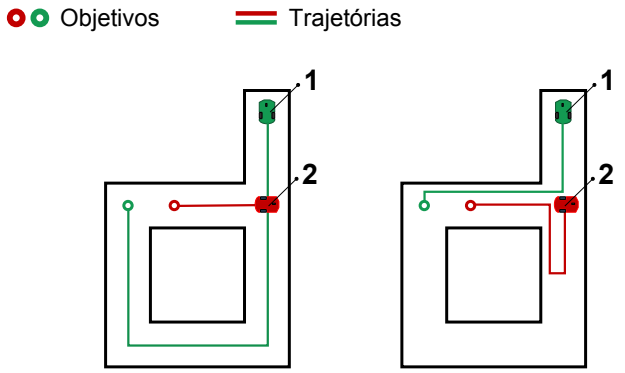
**Completo:** Método capaz de encontrar uma solução para qualquer sistema multi-robôs para o qual existe uma solução

**Ótimo:** Método capaz de encontrar a melhor solução entre as possíveis soluções para um sistema multi-robôs

Abordagens centralizadas realizam uma busca por todo o espaço de estados do sistema combinando as possíveis configurações de todos os robôs e, dessa forma, são capazes de encontrar a solução ótima para o problema sempre que uma solução existir. No entanto, sua complexidade cresce de forma exponencial com o aumento do espaço de estados impondo restrições para sua utilização em sistemas que possuem requisitos temporais críticos.

É comum a utilização de um esquema de prioridades para o tratamento de conflitos em uma abordagem desacoplada. Isso significa que ao encontrar conflitos entre o caminho de dois robôs o caminho de um deles é recalculado considerando o caminho já calculado do robô de maior prioridade. Dessa forma, a dimensão do espaço de estado é reduzida e o problema de busca torna-se tratável. Ao adicionar restrições ao espaço de estados, como a imposição de maior prioridade de um robô sobre outro, essa abordagem torna-se não completa e pode gerar soluções sub-ótimas.

Para sistemas desacoplados a ordem de prioridade no qual os caminhos são computados tem uma forte influência no fato de uma solução ser encontrada ou não e no comprimento (ou custo) do caminho resultante. Para ilustrar a ideia da influencia da ordem de prioridades considere a Figura 28, onde, a partir da posição inicial, os robôs tem que atingir os objetivos indicados. Na Figura 28(a) o caminho do robô 2 é calculado primeiro, então, o caminho do robô 1 inclui um longo desvio até sua meta. Já na Figura 28(b) o caminho do robô 1 é calculado primeiro e, posteriormente, o caminho do robô 2 é calculado levando em conta o caminho do robô de maior prioridade. Dessa forma, obtém-se uma solução muito mais eficiente.




(a) O caminho do robô 2 é calculado primeiro

(b) O caminho do robô 1 é calculado primeiro

Figura 28: Caminhos calculados com diferentes prioridades

No método proposto nesse trabalho, o objetivo é que a exploração do ambiente seja feita de forma eficiente, com o tempo total de exploração reduzido. Por esse motivo, a abordagem selecionada para o planejamento de trajetórias é a abordagem desacoplada com um esquema de prioridades para o tratamento de conflitos. Vale destacar que a abordagem utilizada para o planejamento de trajetória é classificada como desacoplada (em oposição ao método centralizado), que não deve ser confundida com a arquitetura do sistema. A arquitetura utilizada para todo o planejamento de trajetória é classificada como *centralizada*.

O método para cálculo de trajetórias proposto neste trabalho utiliza-se do conceito da abordagem desacoplada com um esquema de prioridades proposto por [Bennewitz 2004]. Em [Bennewitz 2004], no entanto, são utilizados ambientes mais "abertos" impondo menos restrições à movimentação dos robôs e necessitando de um menor grau de coordenação. Este trabalho utiliza a abordagem desacoplada com esquema de prioridades em mapas topológicos, avaliando as soluções através do tempo estimado de execução e ainda propõe uma abordagem nova para coordenação dos múltiplos robôs e tratamento dos conflitos entre suas trajetórias.

### 4.4.1 Método desacoplado proposto

Como mostrado na Figura 15, sempre que uma alocação de tarefas é realizada para todos os robôs um novo cálculo de trajetórias é realizado. Como essa alocação ocorre diversas vezes durante a exploração de um ambiente, o cálculo de trajetórias é também realizado diversas vezes. Dessa forma, a eficiência do sistema não recai simplesmente no fato dos métodos empregados serem ótimos ou não, mas também no tempo de cálculo necessário para cada etapa.

O método para o cálculo de trajetórias proposto nesse trabalho gera uma sequência aleatória de prioridades, definindo a ordem em que os caminhos dos robôs serão calculados. O algoritmo A* de busca em grafos [Nilsson 1982] é, então, utilizado para calcular o caminho ótimo de cada robô para se deslocar da sua posição inicial até seu objetivo. Durante o cálculo das trajetórias dos robôs, eventuais conflitos são identificados e resolvidos através da comparação da trajetória gerada com as trajetórias dos robôs com maior prioridade. O esquema apresentado na Figura 29 ilustra o processo de cálculo de trajetória proposto.

O conjunto de trajetórias gerado a partir de uma ordem de prioridades tem sua eficiência avaliada através de uma estimativa do tempo médio necessário para sua execução. Após essa avaliação, uma nova ordem de prioridades é proposta e o respectivo conjunto de trajetórias é calculado. Os conjuntos de trajetórias são comparados e aquele que apresentar resultado mais eficiente é mantido. O processo é repetido por um número arbitrário de vezes e, ao fim, o conjunto de trajetórias com a menor estimativa de tempo médio é executado pelos robôs.

Em cada nó do grafo é mantida uma lista de controle de acesso. Durante o planejamento é gravada a identificação dos robôs nessa lista representando que em determinado momento da execução da sua trajetória determinados robôs vão acessar o local correspondente. Quando a identificação de um robô de maior prioridade está gravada nessa lista e um robô de menor prioridade também deseja acessá-la, sua identificação é gravada na próxima posição da lista. Essa ordem em que são gravadas as identificações dos robôs representa também a ordem que eles devem acessar o local correspondente no ambiente. Sempre antes de acessar o próximo local previsto no planejamento, deve ser verificado se o primeiro elemento da lista de acesso corresponde à identificação do robô. Caso não corresponda, o robô deve aguardar até que o robô de maior prioridade acesse e deixe o respectivo local.

É difícil prever o tempo que um robô tem que esperar para acessar determinado local do mapa quando a prioridade de acesso é de outro

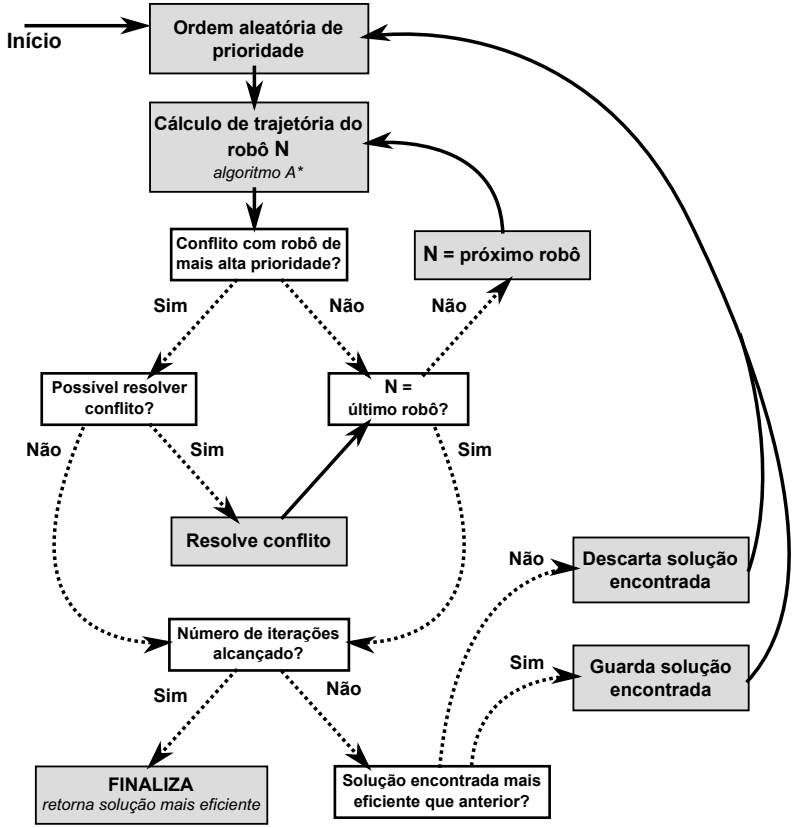



Figura 29: Fluxograma do processo de cálculo de trajetória proposto

robô. Esse tempo de espera depende de inúmeros fatores, como a velocidade do robô, a resolução de conflitos com outros robôs, etc. Neste trabalho, porém, a eficiência do plano de trajetórias gerado é determinada por uma estimativa do tempo feita a partir do número de ações discretas necessárias para que os robôs alcancem seus objetivos. Essas ações discretas correspondem à movimentação entre as *features* do ambiente e, para calcular o número de ações, é considerada a possibilidade de movimentação simultânea e também as esperas de acordo com a prioridade de acesso. Essa estimativa de tempo considera também a distância entre as *features*. A partir do número de ações discretas é possível estimar o tempo necessário para execução das trajetórias e, então,

selecionar a melhor solução entre o conjunto de trajetórias calculadas.

Para ilustrar o processo de análise da eficiência dos planos calculados considere a Figura 30. São mostrados três robôs em suas posições iniciais, e é desejado que os robôs alcancem as posições objetivo indicadas na figura. São também indicadas as trajetórias dos robôs a partir de suas posições iniciais até as posições finais, calculadas desconsiderando a existência dos demais robôs. É possível notar que existem regiões onde há interferência entre trajetórias, como nas células 4 e 6. Essas interferências são os conflitos que devem ser tratados com um esquema de prioridades.

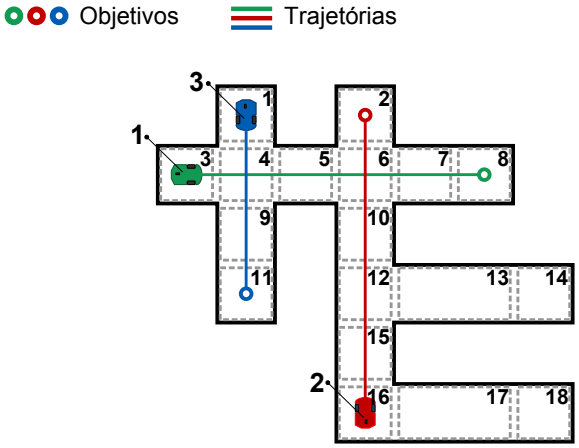



Figura 30: Interferência entre trajetórias de diferentes robôs

Para o exemplo da Figura 30, o processo é iniciado com a geração de uma sequência de prioridade entre os robôs. Na Tabela 1 é apresentada a sequência de células percorridas por cada robô para a ordem de prioridade *Robô 1* > *Robô 2* > *Robô 3*. Enquanto o plano gerado para os robôs 1 e 2 não inclui paradas, já que suas trajetórias não se cruzam, o plano do robô 3 prevê que ele aguarde o robô 1 deixar a célula número 4 antes de ocupá-la. Essa espera pode ser vista na segunda unidade de tempo da linha correspondente ao robô 3.

Como resultado da sequência de prioridade apresentada na Tabela 1, tem-se o robô 1 concluindo sua trajetória em oito unidades de tempo, assim como o robô 2. O robô 3 apresenta um período de espera no seu plano e, portanto, conclui sua trajetória em cinco unidades de tempo. A média de tempo utilizada para conclusão das trajetórias é de aproximadamente $5,67$ unidades de tempo e esse valor é utilizado

| | UNIDADES DE TEMPO | | | | | |
|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** |
| Robô 1 | 3 | 4 | 5 | 6 | 7 | 8 |
| Robô 2 | 16 | 15 | 12 | 10 | 6 | 2 |
| Robô 3 | 1 | // | 4 | 9 | 11 | |

Tabela 1: Sequência de ações discretas para dada prioridade
*Ordem de prioridade: Robô* 1 > *Robô* 2 > *Robô* 3; // - *Espera*

para comparar a eficiência dos planos gerados com outras sequências de prioridade.

Na Tabela 2 são apresentadas as trajetórias para a ordem de prioridades *Robô 2* > *Robô 1* > *Robô 3*. A trajetória do robô 2 não possui parada, por ser o robô de maior prioridade. A trajetória do robô 1 inclui uma espera de duas unidades de tempo, até que o robô 2 deixe a célula 6. Já a trajetória do robô 3 apresenta uma espera de uma unidade de tempo, para que o robô 1 deixe a célula 4. A média de tempo para execução dessa ordem de prioridades é de, aproximadamente, $6,33$ unidades de tempo.

| | UNIDADES DE TEMPO | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| Robô 2 | 16 | 15 | 12 | 10 | 6 | 2 | | |
| Robô 1 | 3 | 4 | 5 | // | // | 6 | 7 | 8 |
| Robô 3 | 1 | // | 4 | 9 | 11 | | | |

Tabela 2: Sequência de ações discretas para dada prioridade
*Ordem de prioridade: Robô* 2 > *Robô* 1 > *Robô* 3; // - *Espera*

Ao fim do processo de geração de trajetórias, o conjunto de trajetórias com menor tempo médio de execução entre as ordens de prioridade avaliadas é executado pelos robôs. No caso do exemplo da Figura 30, considerando as ordens de prioridade apresentadas nas Tabelas 1 e 2, o esquema de prioridades que gerou o conjunto de trajetórias mais eficiente é *Robô 1* > *Robô 2* > *Robô 3*; com a estimativa de tempo médio de execução de $5,67$ unidades de tempo.

Não é sempre, porém, que a inserção de "paradas" e "esperas" na trajetória dos robôs fornece a melhor solução. Muitas vezes, utilizando somente estes recursos não é possível ao menos achar uma solução em que todos os robôs alcancem suas posições finais. Para lidar com esse problema são utilizadas as seguintes heurísticas, apresentadas das se-

ções seguintes:

- Heurística para cálculo de ***caminho alternativo***;
- Heurística para ***resolução de conflitos***

Essas heurísticas estão localizadas, no esquema da Figura 29, no bloco "*Resolve Conflito*", em adição a possibilidade de se inserir esperas na trajetória dos robôs.

#### 4.4.1.1 Caminho alternativo

Nem sempre o caminho ótimo calculado para se deslocar entre dois pontos do mapa é a melhor escolha. Isso porque o tempo que um robô deve esperar para ocupar determinada célula é, muitas vezes, superior ao de se incluir um desvio na sua trajetória.

A heurística de caminho alternativo trata justamente da avaliação da utilização de possíveis rotas alternativas ao invés de esperar a passagem de um robô de mais alta prioridade. Sempre que o cálculo da trajetória de um robô inclui "esperas" uma nova busca é realizada considerando o nó associado à *feature* causadora da espera como um obstáculo. Dessa forma, o algoritmo de busca é forçado a evitar o respectivo nó do grafo.

O nó é considerado ocupado e o plano é calculado; se o resultado for um plano mais eficiente ele é mantido, caso contrário o plano anterior é utilizado. Um número arbitrário de avaliações é realizado, sempre analisando a primeira espera do plano calculado. Quando o bloqueio de um nó implicar na não existência de um caminho até a posição final desejada, o plano anterior também é mantido. O Algoritmo 4 ilustra esse processo.

A Figura 31(a) exemplifica esse procedimento. Os caminhos ótimos de dois robôs a partir das suas posições iniciais até suas posições objetivo são apresentados. Na Tabela 3 são mostradas as trajetórias dos robôs para a ordem de prioridade $Robô\ 2 > Robô\ 1$. A trajetória do robô 2 não inclui esperas por ser o robô com maior prioridade. Já a trajetória do robô 1 inclui uma espera durante três unidades de tempo até que o robô 2 deixe a célula 6.

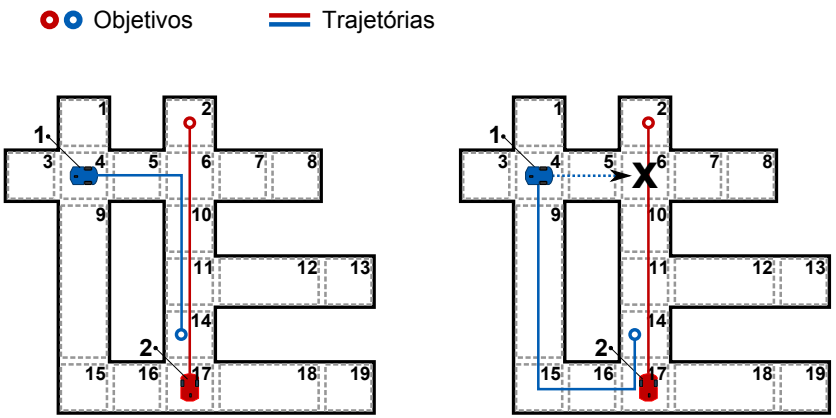
Neste caso, a trajetória do robô 1 é recalculada evitando a célula que causa a espera. O resultado é apresentado na Figura 31(b), na qual a célula 6 é considerada bloqueada para o robô 2 e um caminho alternativo é calculado. Na Tabela 4 é mostrada a sequência de células

**Algoritmo 4:** Heurística de caminho alternativo

```
   input  : NoInicial, NoFinal, Trajetoria
   output: TrajetoriaFinal
 1 β = número arbitrário de iterações
 2 X = 1
 3 for i = 0; i < β do
 4 |   identifica espera nº X da Trajetoria
 5 |   if Trajetoria não possui espera then
 6 |   |   retorna Trajetoria
 7 |   end
 8 |   noBloqueio = nó posterior ao nó de espera
 9 |   noBloqueio ← bloqueado
10 |   novaTrajetoria = calculaTrajetoria(Inicial, Final)
11 |   if novaTrajetoria = vazia then
12 |   |   noBloqueio ← desbloqueado
13 |   |   X = X + 1
14 |   end
15 |   else
16 |   |   if novaTrajetoria < Trajetoria then
17 |   |   |   Trajetoria = novaTrajetoria
18 |   |   |   noBloqueio ← desbloqueado
19 |   |   end
20 |   end
21 end
22 retorna Trajetoria
```

| | **UNIDADES DE TEMPO** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| Robô 2 | 17 | 14 | 11 | 10 | 6 | 2 | | | |
| Robô 1 | 4 | 5 | // | // | // | 6 | 10 | 11 | 14 |

Tabela 3: Sequência de ações discretas para dada prioridade
*Ordem de prioridade: Robô 2 > Robô 1; // - Espera*




(a) Devido a ordem de prioridades, o robô 1 deve aguardar a passagem do robô 2

(b) Célula 6 é "bloqueada" e um caminho alternativo calculado

Figura 31: Caminho alternativo com desvio para evitar interferência

a percorrer e pode-se notar uma diminuição no tempo previsto para execução da trajetória do robô 1.

| | UNIDADES DE TEMPO | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| Robô 2 | 17 | 14 | 11 | 10 | 6 | 2 | | |
| Robô 1 | 4 | 9 | 9 | 9 | 15 | 16 | 17 | 14 |

Tabela 4: Sequência de ações discretas para dada prioridade
*Ordem de prioridade: Robô* 2 > *Robô* 1; // - *Espera*

Outro ponto a se destacar na Figura 31(b) é a inclusão da célula 9 no trajeto do robô. Essa célula se refere a um corredor com comprimento maior do que os outros presentes na figura (células 5 e 12, por exemplo). Percorrer essa célula demanda mais tempo que percorrer as demais e isso deve ser levado em consideração no planejamento da trajetória do robô. Por esse motivo, pode ser visto na Tabela 4 que o robô 1 precisa de três unidades de tempo para percorrer a célula 9. O tempo previsto para percorrer as células associadas a corredores é, então, proporcional ao seu comprimento.

A heurística de *caminho alternativo* é opcional ao método de cálculo de trajetória. Ela visa encontrar um conjunto de caminhos

cuja execução seja mais rápida do que a simples inserção de esperas e paradas. Porém, a sua não utilização não implica no comprometimento do método de planejamento de trajetória proposto.

#### 4.4.1.2 Resolução de conflitos

Existem situações em que somente a utilização de uma ordem de prioridades para o cálculo de trajetórias e o controle da ocupação dos nós do grafo, como apresentado, não é suficiente para que todos os robôs cheguem em suas posições finais desejadas. São duas as situações principais que, se não forem tratadas de forma diferenciada, podem causar bloqueios no sistema. Essas situações são as seguintes:

1. A posição final de um robô de maior prioridade está no caminho de um robô de menor prioridade;
2. A posição inicial de um robô de menor prioridade está no caminho calculado para um robô de maior prioridade.

O primeiro problema listado ocorre, pois o robô de menor prioridade deve aguardar a passagem do robô de maior prioridade por todos os nós do grafo correspondentes a sua trajetória, para só então poder ocupar qualquer um dos nós pertencente a trajetória de ambos. Chegando o robô de maior prioridade a sua posição final, o outro robô espera indefinidamente sua saída, gerando um bloqueio no sistema.

Para um planejador de trajetória de sistemas multi-robôs, o qual o objetivo é simplesmente alcançar pontos do mapa, esse é um problema bastante relevante. Porém, para o objetivo de exploração de ambientes desconhecidos, o problema é contornado devido a sua natureza. Os objetivos da exploração são alcançar as *features* com fronteiras associadas no ambiente e, após alcançá-las, avançar sobre alguma das regiões ainda desconhecidas. Dessa forma, os robôs não ficam parados sobre o seu destino final impedindo a execução da trajetória dos demais.

Para utilização do planejador em um problema diferente da exploração de ambientes desconhecidos, esse problema poderia ser tratado através da mesma solução adotada para a heurística de *caminho alternativo*. A célula "conflituosa" deveria ser marcada como um obstáculo e outros caminhos calculados. Caso não exista uma solução, a ordem de prioridade utilizada entre os robôs envolvidos não deve mais fazer parte do espaço de busca.

O problema pode ser ilustrado através da Figura 32. Para um objetivo de alcançar as células indicadas, o robô 1 ficaria esperando

indefinidamente a saída do robô 2 da respectiva célula.

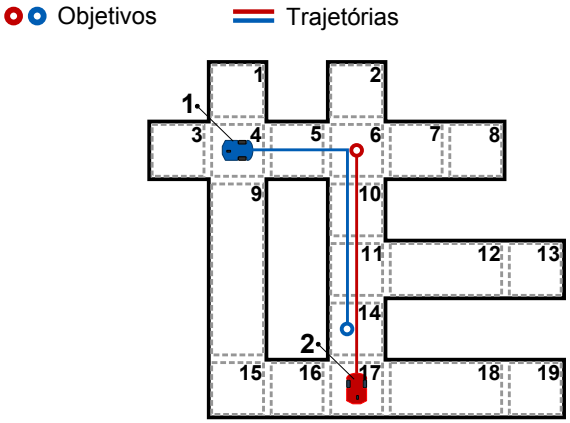



Figura 32: Robô 2 com maior prioridade causando bloqueio

O segundo problema é mais relevante quando o objetivo do planejador é a exploração de um ambiente desconhecido. Quando um robô de menor prioridade inicia sua trajetória sobre o caminho calculado para um robô de maior prioridade, este possui a prioridade de acesso ao nó em questão. Com o robô de menor prioridade posicionado neste mesmo nó o sistema seria levado a uma situação de bloqueio.

O robô de menor prioridade deve sair da trajetória do robô de maior prioridade e deve ter prioridade no acesso à sua posição inicial. Para resolver esse conflito a trajetória do robô de menor prioridade é, então, dividida em duas porções: a primeira delas com prioridade maior que a do outro robô e a segunda com a mesma prioridade inicialmente dada ao robô. A primeira porção da trajetória tem como objetivo sair do caminho do robô de maior prioridade, selecionando para isso uma célula não pertencente à trajetória do mesmo e que seja próxima à trajetória do próprio robô. A segunda porção da trajetória tem como objetivo levar o robô até sua posição final. O Algoritmo 5 apresenta esse procedimento.

A Figura 33 apresenta um exemplo de situação na qual a posição inicial de um robô de menor prioridade está sobre o caminho de um robô de maior prioridade, quando a ordem de prioridades *Robô* 2 > *Robô* 1 > *Robô* 3 é utilizada. Nesse exemplo, o robô 3 parte da célula 6, que faz parte da trajetória dos demais robôs. Se o cálculo de trajetórias for feito sem um cuidado especial com essa situação, o robô 2 teria

**Algoritmo 5:** Heurística de resolução de conflito

```
   input : OrdemPrioridade
1  for i = 0 : OrdemPrioridade do
2  |  trajetoriaRobo(i) = Calcula trajetória do robô com
   |  prioridade = i
3  |  while trajetoriaRobo(i) possui conflito com robô de
   |  prioridade < i do
4  |  |  j =prioridade do robô conflituoso
5  |  |  celulaX =busca célula ∉ trajetoriaRobo(j)
6  |  |  trajetoriaRobo(j − 1) =Calcula trajetória robô i da
   |  |  célula inicial → celulaX
7  |  |  trajetoriaRobo(i) =Calcula trajetória robô i da
   |  |  célula celulaX → final
8  |  end
9  end
```

prioridade de acesso à célula 6, porém, com o robô 3 já localizado nela, ele não poderia seguir sua trajetória, levando o sistema a uma situação de bloqueio.

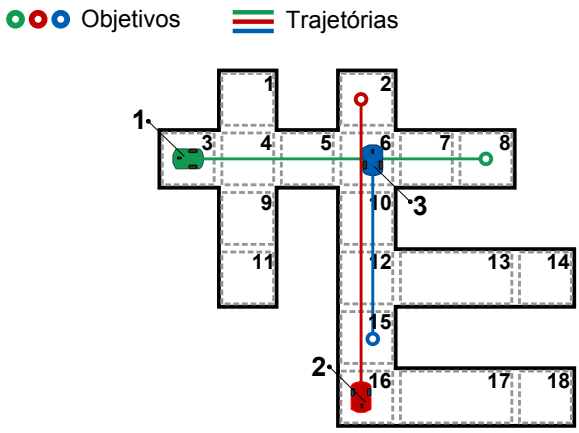



Figura 33: Posição inicial do robô sobre o caminho dos demais

Para resolver esse conflito, o robô 3 deve sair da trajetória do robô de maior prioridade e deve ter prioridade no acesso à sua posição inicial. Como o primeiro conflito ocorre com o robô 2, a trajetória do

robô 3 é dividida em duas porções: a primeira delas com prioridade maior que a do robô 2 e a segunda com a prioridade inicialmente dada ao robô 3. A primeira porção da trajetória tem como objetivo sair do caminho do robô 2, selecionando para isso uma célula não pertencente à trajetória do mesmo e que seja próxima (preferencialmente pertencente) à trajetória do próprio robô 3. A segunda porção da trajetória tem como objetivo levar o robô até sua posição final.

Na Tabela 5 é mostrado o conjunto de trajetórias após a resolução do conflito entre os robôs 2 e 3. Com maior prioridade está a primeira porção da trajetória do robô 3, levando o robô para a célula 7 com intuito de sair do caminho do robô 2. Na sequência temos, nesta ordem, os robôs 2 e 1 com as trajetórias até seus objetivos. Por último, temos a segunda porção da trajetória do robô 3, que agora parte da célula 7. Um novo conflito é identificado entre os robôs 1 e 3, pois o robô 3 parte da célula 7 que está contida na trajetória do robô de maior prioridade.

| | **UNIDADES DE TEMPO** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| Robô $3^{i}$ | 6 | 7 | | | | | | |
| Robô 2 | 16 | 15 | 12 | 10 | 6 | 2 | | |
| Robô 1 | 3 | 4 | 5 | // | // | 6 | 7 | 8 |
| Robô $3^{ii}$ | | 7 | | | | | | |

Tabela 5: Divisão da trajetória do robô 3 para resolução de conflito
*Ordem de prioridade: Robô* 2 > *Robô* 1 > *Robô* 3; // - *Espera*

Para resolver esse conflito a trajetória do robô 3 é, novamente, dividida em duas porções. A primeira delas, com prioridade maior que a do robô 1 tem como objetivo a saída do robô 3 do caminho do robô 1. Para isso, é selecionada uma célula que não pertença à trajetória do robô 1 e que seja próxima à trajetória do robô 3. A célula 10 é selecionada e um caminho da célula 7 até a célula 10 é calculado, respeitando as trajetórias dos robôs de maior prioridade.

O resultado dessa nova resolução de conflitos é apresentado na Tabela 6. A trajetória do robô 3 é realizada em três etapas com prioridades diferentes, a primeira delas cedendo passagem ao robô 2, a segunda cedendo passagem ao robô 1 e a terceira deslocando-se até seu objetivo. O resultado final não apresenta conflitos e pode ser executado pelos robôs.

| | UNIDADES DE TEMPO | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Ordem** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| Robô $3^{i}$ | 6 | 7 | | | | | | | |
| Robô 2 | 16 | 15 | 12 | 10 | 6 | 2 | | | |
| Robô $3^{ii}$ | | 7 | // | // | // | 6 | 10 | | |
| Robô 1 | 3 | 4 | 5 | // | // | // | 6 | 7 | 8 |
| Robô $3^{iii}$ | | | | | | | 10 | 12 | 15 |

Tabela 6: Nova divisão da trajetória do robô 3 para resolução de conflito
*Ordem de prioridade: Robô* 2 > *Robô* 1 > *Robô* 3; // - *Espera*

A heurística de *resolução de conflitos* evita situações de bloqueio no sistema, trabalhando com a ordem de prioridades dos robôs quando existe a impossibilidade de execução dos caminhos calculados. Por isso, sua utilização é obrigatória na etapa do cálculo de trajetórias.

## 4.5 CONCLUSÃO

Neste capítulo foi apresentado o método proposto para a exploração de ambientes desconhecidos com sistemas multi-robôs, passando pelas duas principais etapas da resolução desse problema: o mapeamento e o planejamento de trajetória.

Para o mapeamento foi utilizada uma abordagem topológica, na qual características métricas do ambiente são incorporadas, facilitando assim a resolução de problemas recorrentes na abordagem topológica, como a de determinação de equivalência de regiões diferentes do ambiente. As estruturas do ambiente selecionadas para serem transformadas em nós do grafo são as *features*, nesse trabalho classificadas como corredores, esquinas, junções, cruzamentos e becos-sem-saída. O mapa gerado é, então, um grafo não orientado onde os corredores são os arcos e as demais *features* são os nós. A cada nó do grafo são associadas as coordenadas correspondentes à localização aproximada de uma *feature* em relação às demais.

O planejamento de trajetória foi dividido em duas etapas: a alocação de tarefas e o cálculo da trajetória. A etapa da alocação de tarefas é responsável por determinar qual ponto do mapa deverá ser explorado por cada robô, dada a distribuição dos robôs no mapa parcial já construído.

As técnicas empregadas nessa etapa influenciam diretamente no tempo total de exploração do ambiente. Com intuito de diminuir o nú-

mero de interferência entre as trajetórias dos robôs e da passagem dos mesmos por regiões já exploradas do ambiente, executando trabalho redundante, foram propostas três heurísticas para lidar com a alocação de tarefas. Cada uma das heurísticas contribui com um "custo" que somados determinam o custo total para que um robô alcance determinada fronteira. O papel do alocador é buscar uma distribuição de tarefas entre os robôs com menor custo total. As heurísticas propostas são:

**Menor distância** Designar a cada robô a exploração da fronteira mais próxima a ele, evitando que os robôs tenham que atravessar caminhos já explorados para retornar a essas fronteiras futuramente;

**Distribuição dos robôs** Distribuir os robôs pelo mapa, diminuindo o grau de interferência entre os mesmos que provocam a inclusão de desvios e paradas de espera em suas trajetórias;

**Fechamento de loop** Priorização das exploração de fronteiras cuja *feature* associada a ela seja, potencialmente, finalizada em outra fronteira conhecida

A etapa do cálculo de trajetória realiza o cálculo do caminho que um robô deve percorrer para realizar a tarefa designada. A abordagem selecionada para o método proposto é a abordagem desacoplada com um esquema de prioridades para o tratamento de conflitos. Nesse método, uma sequência aleatória de prioridades entre os robôs é gerada definindo a ordem em que os caminhos dos robôs serão calculados. O algoritmo A* de busca em grafos é, então, utilizado para calcular o caminho ótimo de cada robô para se deslocar de sua posição inicial até seu objetivo. Durante o cálculo das trajetórias dos robôs, eventuais conflitos são identificados e resolvidos através da comparação da trajetória gerada com as trajetórias dos robôs de maior prioridade.

Muitas vezes, as soluções encontradas utilizando somente a sequência de prioridades para tratamento de conflitos não é eficiente ou mesmo não encontra um caminho para que todos os robôs alcancem suas posições finais. Para lidar com essas situações duas heurísticas foram propostas para a etapa do cálculo de trajetória:

**Caminho alternativo** A heurística trata da avaliação de possíveis rotas alternativas ao invés de esperar a passagem de um robô de mais alta prioridade, evitando a *feature* conflituosa e buscando soluções mais eficientes;

**Resolução de conflitos** A heurística insere desvios nas trajetórias dos robôs para que robôs de maior prioridade possam alcançar suas posições finais liberando a passagem de robôs de menor prioridade, evitando bloqueios no sistema

O capítulo seguinte trata de aspectos da implementação do sistema em ambiente de simulação e realiza diversos experimentos variando ambiente, número de robôs e técnica empregada. O método de exploração de ambientes desconhecidos proposto neste capítulo é, então, analisado e avaliado através dos resultados dos experimentos.

# 5 RESULTADOS EXPERIMENTAIS

Para consolidar e validar o método de exploração de ambientes desconhecidos com sistemas multi-robôs proposto nessa dissertação, uma série de experimentos são apresentados nesse capítulo. Esses experimentos são realizados utilizando diversos ambientes estruturados diferentes e variados números de robôs. Características do planejador de trajetórias são também exploradas, habilitando-se gradativamente as heurísticas do alocador de tarefas, por exemplo.

Durante a elaboração deste trabalho foram realizados experimentos em pequenos protótipos de robôs móveis, desenvolvidos pelo autor e pelo grupo de pesquisa o qual esse trabalho está inserido. Os experimentos, entretanto, geraram apenas resultados relacionados à navegação dos robôs por ambientes desconhecidos com algoritmos de exploração mais simples. Até a conclusão dessa dissertação, os métodos aqui propostos não haviam sido implementados nos robôs, e os resultados apresentados foram obtidos através de simulação.

Alguns dos principais pontos que se busca responder através dos experimentos estão relacionados à praticidade e eficiência do método. Entende-se como praticidade o fato do método proposto conseguir solucionar o problema de exploração, fornecendo um mapa compatível com o ambiente em questão e não levando o sistema a situações de bloqueio.

Já a eficiência está relacionada ao fato das heurísticas propostas para o alocador de tarefas fornecerem soluções significativamente mais rápidas para a exploração, se comparadas com uma alocação realizada de forma aleatória. Busca-se também identificar se há relação entre a eficiência das heurísticas propostas com as características do ambiente e o número de robôs executando a exploração.

Para os experimentos é adotado um ambiente estruturado, como apresentado na Seção 2.4. O ambiente em questão é *indoor*, sem nenhum sistema de posicionamento global disponível e estruturado: composto unicamente por paredes paralelas e perpendiculares umas às outras. Considera-se um grupo de robôs móveis com rodas, equipados com sonares na região frontal e *encoders*, capazes de se comunicar com uma estação central responsável pelo mapeamento e planejamento das trajetórias. Os experimentos realizados nessa dissertação utilizam o ambiente *Player/Stage* para simulação e obtenção de resultados.

## 5.1 AMBIENTE DE SIMULAÇÃO *PLAYER/STAGE*

O projeto *Player/Stage* teve seu desenvolvimento iniciado em 1999 na *University of Southern California* - EUA para suprir uma necessidade interna de interface e simulação para sistemas multi-robôs. Por ser um projeto de código aberto e livre distribuição, fornecer um controlador e simulador de grande compatibilidade e por ser independente de arquiteturas de programação, *Player/Stage* se popularizou bastante entre os centros de desenvolvimento de pesquisas ligadas à robótica e, desde então, tem sido utilizado, desenvolvido e expandido por pesquisadores e universidades ao redor de todo o mundo [Gerkey, Vaughan e Howard 2003].

*Player* fornece uma interface simples de controle de robôs, através de uma rede com funcionamento baseado no modelo servidor/cliente. O servidor faz interface com o robô e com outros sensores, obtendo dados, enviando-os para o cliente e recebendo instruções do cliente para controle do robô e dos sensores. O cliente é o programa que controla efetivamente o robô, sendo responsável por obter dados do servidor e enviar instruções para a realização de determinada tarefa. Toda comunicação entre cliente e servidor é realizada através de TCP/IP.

A arquitetura cliente/servidor permite grande versatilidade no controle de robôs e sensores, de forma que um cliente pode controlar diversos servidores e diferentes clientes podem controlar diferentes sensores e atuadores de um mesmo robô. *Player* foi projetado para ser compatível com diversas linguagens de programação. Neste trabalho é utilizada a linguagem Java, através da biblioteca *javaclient3* [Ostergard 2001].

*Stage* é um simulador de robôs para ambientes bidimensionais compatível com *Player*. Múltiplos robôs e sensores podem ser simulados simultaneamente, controlados por um ou mais clientes. Esses dispositivos são acessados pelo *Player* como se fossem o *hardware* real dos robôs. A Figura 34 exibe uma tela de simulação do *Stage*.

## 5.2 EXPERIMENTOS

Os experimentos apresentados nessa seção utilizam como medida da eficiência dos métodos empregados o tempo total de exploração dos ambientes desconhecidos. Em todos os casos, os testes foram repetidos em torno de dez vezes com as mesmas configurações (mesmo ambiente, ponto inicial, etc) sendo a média do tempo de execução apresentado

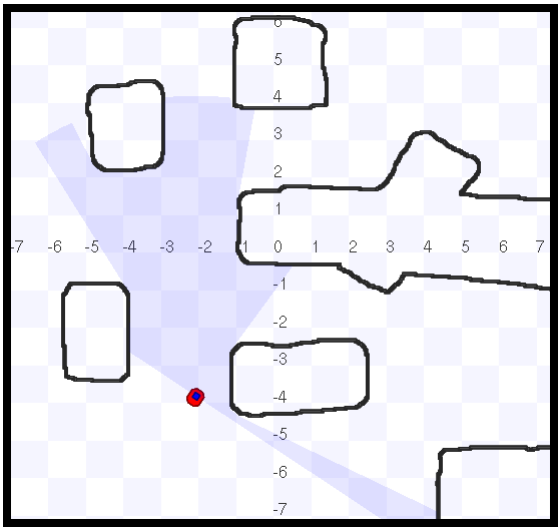



Figura 34: Ambiente *Stage*

nos gráficos, juntamente com o intervalo de confiança de 95%.

Essa repetição é feita devido à característica não determinística do problema que pode apresentar soluções diferentes a cada execução. A alocação de tarefas, por exemplo, é realizada a partir de uma função custo que seleciona a fronteira "mais vantajosa" a ser explorada. Muitas vezes, porém, essa função custo pode retornar valores idênticos para duas tarefas e, então, uma decisão é tomada de forma aleatória. Outro ponto que pode tornar o resultado de duas execuções diferentes é o fato de que diferenças na ordem de poucos segundos na movimentação dos robôs podem definir qual deles irá "descobrir" uma nova *feature*. Se em uma execução um robô descobre essa *feature* e na execução seguinte um outro robô descobre essa mesma *feature*, teremos soluções diferentes.

Como apresentado na Seção 2.4, é assumido que os robôs possuem, no início da exploração, uma referência global, fazendo com que seus sistemas de coordenadas sejam os mesmos e que as informações coletadas do ambiente sejam diretamente incorporadas a um único mapa global. Nos experimentos apresentados nessa seção os robôs partem todos de um mesmo ponto inicial, porém, em instantes de tempo diferentes. O primeiro robô inicia a exploração e, assim que descoberta fronteiras suficientes para também alocar uma tarefa ao próximo robô, o segundo inicia a exploração. Quando já existem tarefas suficientes

para todos os robôs, um pequeno atraso é utilizado entre a saída de cada dois robôs. Entretanto, não foi realizado um estudo do impacto da utilização desse método de inicialização no tempo resultante para a exploração.

Os experimentos utilizam como base de comparação a alocação realizada *aleatoriamente*. Essa alocação ocorre selecionando de forma aleatória a tarefa a ser executada. Porém, quando o robô já está posicionado em uma *feature* a qual possui fronteiras a serem exploradas, a tarefa é selecionada entre as fronteiras em questão, evitando "idas" e vindas do robô. Nesse método de alocação são também utilizados os mesmos métodos de cálculo de trajetória apresentados na Seção 4.4.

São apresentados três experimentos. Os dois primeiros avaliam o uso das heurísticas para alocação de tarefas apresentadas na Seção 4.3.2:

1. Menor distância;
2. Distribuição dos robôs;
3. Fechamento de *loop*

O terceiro experimento avalia o efeito do aumento do número de robôs na exploração de um mesmo ambiente, comparando o efeito da utilização de heurísticas na alocação de tarefas com uma alocação realizada de forma aleatória.

### 5.2.1 Heurísticas isoladas

Esse experimento tem como objetivo avaliar a contribuição de cada heurística na melhora da eficiência do sistema, comparando com uma alocação realizada aleatoriamente. Para isso, simulações foram realizadas em vários ambientes diferentes, porém de extensão aproximada. A Figura 35 apresenta alguns dos ambientes utilizados nas simulações.

As simulações foram realizadas variando o número de robôs de 1 até 4. Os resultados são apresentados no gráfico da Figura 36, no qual as seguintes heurísticas são utilizadas:

- Alocação realizada aleatoriamente;
- Alocação realizada utilizando a heurística o fechamento de *loop*;
- Alocação realizada utilizando a heurística de menor distância;

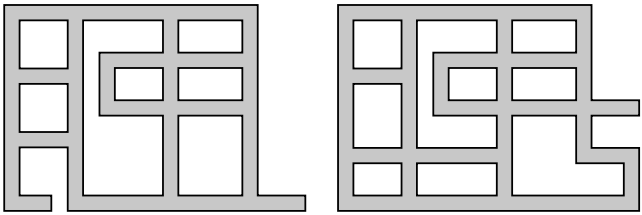


Figura 35: Ambientes utilizados na avaliação de heurísticas

- Alocação realizada utilizando a heurística de distribuição dos robôs

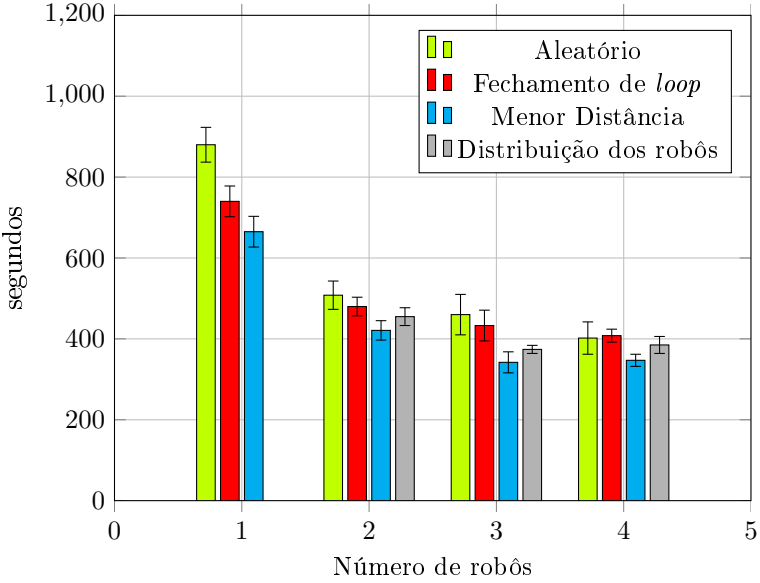



Figura 36: Variação do número de robôs na utilização isolada de heurísticas

É possível observar que, para o caso de apenas um robô, as heurísticas de *fechamento de loop* e de *menor distância* apresentam uma diminuição significativa do tempo de exploração. A heurística de *distribuição dos robôs* não é aplicada para o caso de um robô pois os custos são calculados pela distância de fronteiras alocadas a outros

robôs, e, como não existem outros robôs na simulação, os custos seriam sempre nulos. Vale destacar que a aplicação da heurística de *menor distância* apresenta uma maior redução no tempo de exploração que a heurística de *fechamento de loop*.

Para o caso de dois robôs, todas as três heurísticas apresentam redução no tempo de exploração, porém mais sutis que no caso de um robô. A heurística que apresenta maior redução de tempo é a de *menor distância*, seguida pela de *distribuição dos robôs* e, por último, *fechamento de loop*. Um comportamento semelhante é observado no caso de três robôs.

Para o caso de quatro robôs, as heurísticas de *menor distância* e *distribuição dos robôs* apresentam reduções no tempo de exploração. Um ponto interessante é que a heurística de *fechamento de loop* apresenta um pequeno aumento no tempo. Isso acontece, pois, utilizando apenas a heurística de *fechamento de loop* os robôs tendem a explorar fronteiras que possivelmente estejam conectadas a *features* já conhecidas do ambiente. Como não está sendo utilizada a heurística de *distribuição dos robôs*, os robôs não tendem a se afastar uns dos outros. Comparando com os outros casos (1, 2 e 3 robôs) temos também uma maior "densidade" de robôs, já que os ambientes de simulação possuem a mesma extensão.

Conclui-se, então, que a heurística de *fechamento de loop* aplicada sozinha acaba promovendo mais encontros entre os robôs, causando um aumento de tempo que supera a economia de se fechar *loops*. Esse aumento do número de encontros é causado pois em um ambiente mais "denso" de robôs o alocador de tarefas acaba fazendo com que dois robôs tentem fechar um mesmo *loop* a partir de direções opostas. Assim, a aplicação conjunta da heurística de *distribuição dos robôs* evitaria esse efeito. Na Seção 5.2.2 são avaliadas as heurísticas aplicadas de forma combinada.

Com base nos resultados obtidos são, então, definidos os pesos das constantes $\alpha$, $\beta$ e $\gamma$ da Equação 4.1. Definiram-se com valores mais elevado as constantes relacionadas às heurísticas que mais contribuem com a diminuição do tempo de exploração. Dessa forma $\alpha = 0.5$ (heurística de menor distância), $\beta = 0.3$ (heurística de distribuição dos robôs) e $\gamma = 0.2$ (heurítica de fechamento de *loops*).

### 5.2.2 Combinação de heurísticas

O experimento de combinação das heurísticas busca avaliar os efeitos da utilização de mais de uma heurística simultaneamente na exploração de ambientes desconhecidos. Para isso, uma série de simulações foi realizada, variando o número de robôs e habilitando gradativamente as heurísticas propostas para a alocação de tarefas. Exemplos dos ambientes utilizados para a simulação são mostrados na Figura 35.

O resultado das simulações é apresentado no gráfico da Figura 37, no qual são apresentadas quatro conjuntos de barras correspondendo a exploração realizada variando-se o número de robôs. Cada barra corresponde a aplicação de uma ou um conjunto de heurísticas, da seguinte forma:

- Alocação aleatória;
- Heurística de fechamento de *loop*;
- Heurística de fechamento de *loop* e de menor distância;
- Heurística de fechamento de *loop*, de menor distância e de distribuição do robôs

Para o primeiro conjunto de barras, de exploração com apenas um robô, pode-se notar uma diminuição significativa do tempo de exploração com a utilização da heurística de *fechamento de loop* em comparação com a alocação realizada de forma aleatória. Adicionando a heurística de *menor distância*, uma redução de igual grandeza pode ser percebida.

Para a exploração realizada com dois e três robôs um comportamento semelhante é observado. Uma redução no tempo de exploração habilitando a heurística de *fechamento de loop* e uma redução semelhante ao se adicionar a heurística de *menor distância*. Porém, nota-se uma maior redução do tempo de exploração ao se adicionar a heurística de *distribuição dos robôs*.

No experimento apresentado na Seção 5.2.1 foi constatado que, aplicada isoladamente, a heurística de *distribuição dos robôs* não é a que fornece uma maior redução de tempo. Já para o experimento de heurísticas combinadas o resultado é diferente, mostrando que a combinação de heurísticas traz ganhos para o método de alocação. Com a combinação das heurísticas de *fechamento de loop* e *distribuição dos robôs*, por exemplo, são evitadas as situações em que dois robôs tentam fechar um mesmo *loop* em direções opostas.

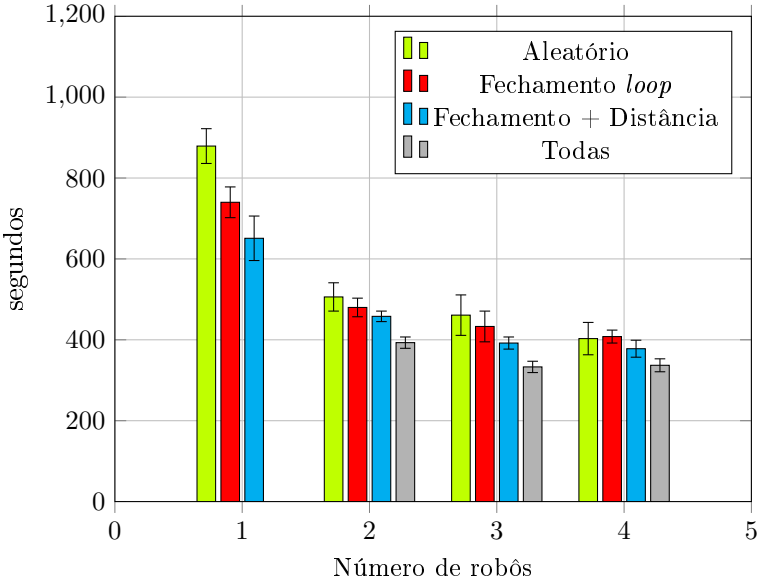



Figura 37: Variação do número de robôs na utilização combinada de heurísticas

Vale também destacar que para a exploração realizada com a alocação aleatória existe uma pequena diferença de tempo para a exploração realizada com dois e com três robôs, identificando uma possível subutilização desse robô adicional. Já para a exploração utilizando todas as heurísticas propostas essa diferença de tempo é maior.

No último conjunto de barras, de exploração com quatro robôs, apesar de existir uma redução do tempo de exploração com a utilização das heurísticas nota-se que ela é mais sutil. No último ponto, utilizando todas as heurísticas propostas, o tempo de exploração é muito próximo para os casos de três e quatro robôs. Esse fato sugere uma ideia de "saturação" do ambiente quando a densidade de robôs em um ambiente é elevada. A Seção 5.2.3 trata desse aspecto com mais detalhes.

### 5.2.3 Variação do número de robôs

O objetivo desse experimento é avaliar a influencia da variação do número de robôs no tempo de exploração de um ambiente desconhe-

cido, observando o efeito da utilização das heurísticas propostas para a alocação de tarefas. Para isso, é proposto um ambiente mais "aberto", onde é possível que os robôs se distribuam durante a exploração. O ambiente é apresentado na Figura 38.

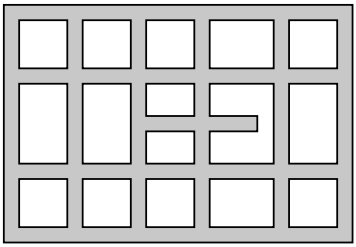


Figura 38: Ambiente com aproximadamente 70 *features*

O experimento foi realizado executando diversas explorações no ambiente, com alocação de tarefas aleatória e também com todas as heurísticas propostas simultaneamente. O gráfico da Figura 39 apresenta os resultados.

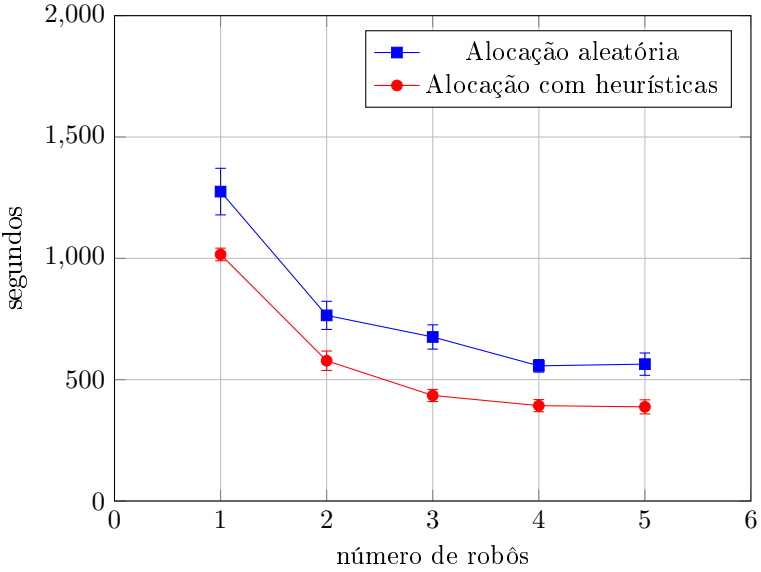



Figura 39: Tempo de exploração - ambiente pequeno

É possível observar que, para esse caso, a variação do número de robôs causa uma redução do tempo de exploração em uma relação próxima a $tempo = e^{-nRobo}$. A partir de quatro robôs o ambiente "satura" e o aumento do número de robôs não aumenta mais a eficiência do sistema. A curva com alocação realizada através das heurísticas propostas apresenta um comportamento similar ao de alocação aleatória, porém, adicionada de uma constante que a torna mais eficiente em todas as situações.

Com intuito de comprovar que o efeito observado realmente se trata de uma "saturação" do ambiente, ligada à relação entre número de robôs e dimensão do ambiente, um novo experimento é proposto em um ambiente com características semelhantes ao do experimento anterior, porém, com maiores dimensões. O ambiente em questão é apresentado na Figura 40.

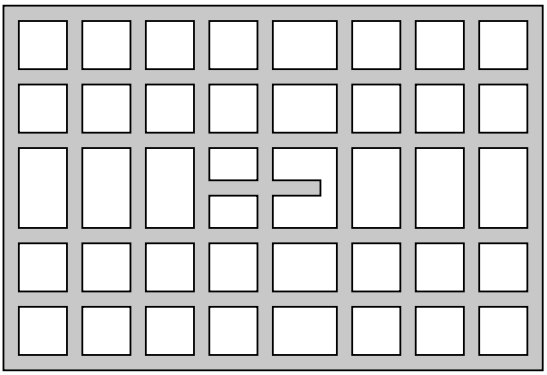


Figura 40: Ambiente com aproximadamente 150 *features*

A exploração do ambiente é realizada diversas vezes, variando o número de robôs de 1 a 8, e realizando a alocação de forma aleatória e também com as heurísticas propostas.

Os resultados, apresentados no gráfico da Figura 41, possuem características semelhantes ao experimento com ambiente de menor dimensão, com uma grande redução ao se adicionar os primeiros robôs e uma redução quase nula adicionando os últimos. Aumentando gradativamente o número de robôs é possível observar uma redução no tempo exploração até o sexto robô. Depois, o tempo de exploração permanece aproximadamente constante mesmo com mais robôs executando a exploração.

O experimento em ambiente reduzido apresentou melhora no

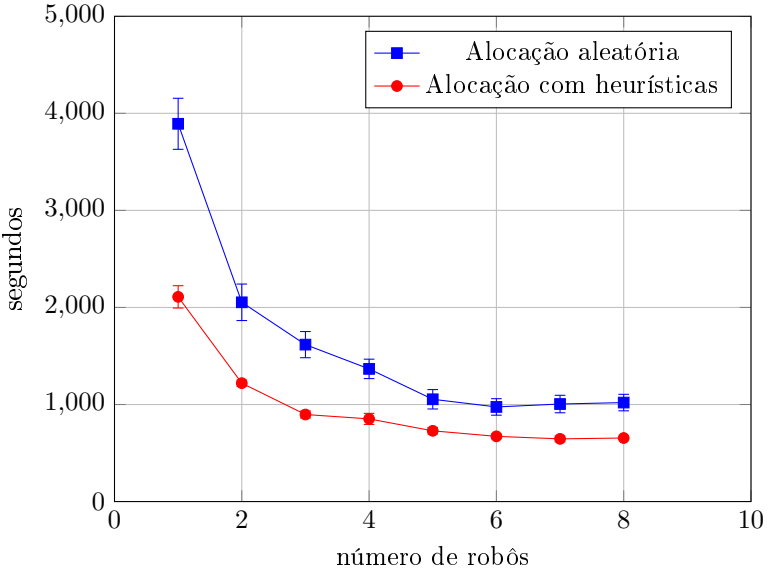



Figura 41: Tempo de exploração - ambiente grande

tempo de exploração até o quarto robô e o experimento em ambiente de maior dimensão até o sexto. Com esses dados conclui-se que existe sim uma "saturação" do ambiente ligada a uma relação entre o número de robôs e o tamanho do ambiente sendo explorado.

Outro ponto relevante que pode ser concluído através desse experimento está ligado à complexidade computacional. O método proposto para exploração de ambientes desconhecidos realiza o planejamento de trajetórias inúmeras vezes durante uma exploração. Mesmo nos experimentos realizados com oito robôs explorando o ambiente de maior dimensão, não foram observadas pausas significativas para realização do planejamento de trajetórias.

Numa abordagem centralizada, onde os espaços de estados de cada robô são combinados para, então, ser realizada uma busca, o número de estados seria próximo a $\textit{número de posições discretas}^{nRobos} = 150^8 \approx 2,56 \times 10^{17}$. Esse número de estados torna impraticável a realização de uma busca com algoritmos clássicos em um *hardware* similar ao que foi utilizado para a execução dos experimentos, evidenciando assim a baixa complexidade computacional do método proposto em comparação a métodos centralizados clássicos. Não foram realizadas

comparações com outros métodos específicos de exploração; a análise em relação à complexidade computacional é feita apenas para destacar diferenças entre as abordagens desacoplada e centralizada.

## 5.3 CONCLUSÃO

Nesse capítulo foram apresentados experimentos aplicando o método de exploração de ambientes desconhecidos com sistemas multi-robôs proposto ao longo desse trabalho. O método proposto foi empregado implementando todas as etapas envolvidas do processo de exploração, como o mapeamento e o planejamento de trajetórias.

Esses experimentos tiveram como objetivo avaliar o comportamento do método para variados ambientes e o efeito do aumento do número de robôs, destacando aspectos relacionados à praticidade do método e a sua eficiência.

Foram propostos três experimentos:

**Heurísticas isoladas** Avaliação da contribuição de cada heurística isoladamente para a redução do tempo de exploração;

**Combinação de heurísticas** Utilização de mais de uma heurística simultaneamente e avaliação da combinação das mesmas na redução do tempo de exploração;

**Variação do número de robôs** Avaliação do efeito da variação do número de robôs na exploração de um mesmo ambientes.

Como resultado dos experimentos conclui-se que o método proposto apresenta uma solução de baixa complexidade computacional para o problema de exploração de ambientes desconhecidos com sistemas multi-robôs. O método é capaz de resolver o problema mesmo quando o ambiente possui um número elevado de *features* e o sistema utilizado é formado por um conjunto relativamente grande de robôs, sem levar o coordenador a um estado de explosão combinatória nem de necessitar de intervalos significativamente grandes de tempo para os cálculos de trajetórias.

As heurísticas propostas para a alocação de tarefas contribuem para a redução do tempo de exploração mesmo quando utilizadas isoladamente. Isso mostra que elas atendem aos seus objetivos iniciais, de diminuir o número de interferência entre os robôs e de diminuir a passagem dos robôs por regiões já exploradas do ambiente.

A combinação dessas heurísticas apresenta resultados significativamente melhores que uma alocação realizada de forma aleatória.

No último experimento buscou-se estudar o efeito do aumento da variação do número de robôs na exploração de um ambiente. Foi concluído que existe uma "saturação" do ambiente para a qual o aumento do número de robôs não resulta numa diminuição do tempo de exploração e também identificada a curva que mostra a relação entre o número de robôs e o tempo de exploração.

# 6 CONCLUSÕES E PERSPECTIVAS

Neste trabalho foram apresentados os principais desafios na resolução do problema de exploração de ambientes desconhecidos e as características particulares desse problema quando tratado com sistemas multi-robôs. Os sistemas multi-robôs apesar de potencialmente muito eficientes, possuem uma série de questões a serem consideradas para um real aproveitamento de suas funções. Entre essas questões, as que foram foco desse trabalho se concentram na alocação de tarefas e coordenação dos múltiplos robôs.

Um método de exploração de ambientes desconhecidos estruturados com sistemas multi-robôs foi proposto, trazendo abordagens para a realização do mapeamento e do planejamento de trajetórias. Para o mapeamento, *features* são extraídas do ambiente e incorporadas em um grafo não orientado, no qual os corredores são os arcos e demais *features* são os nós. Nesse mapa topológico são incorporados dados métricos obtidos durante a exploração, resultando em uma representação compacta e de rápido processamento, porém com dados da localização dos nós auxiliando na resolução de problemas, como a determinação de equivalência.

Para o planejamento de trajetória uma abordagem desacoplada com um esquema prioridades foi proposta. Nessa abordagem o caminho de cada robô é calculado separadamente e conflitos em suas trajetórias são tratados posteriormente. A abordagem proposta apresenta uma alternativa à abordagens centralizas, na qual o espaço de estados de todos os robôs são combinados e um algoritmo de busca é utilizado para encontrar uma solução.

A etapa de planejamento de trajetórias foi divida em outras duas etapas: a alocação de tarefas e o cálculo da trajetória. Na alocação de tarefa, heurísticas foram propostas com intuito principal de reduzir o tempo total de exploração, buscando para isso distribuir os robôs pelo ambiente e também evitar a passagem dos robôs por regiões já exploradas. Para o cálculo de trajetórias foram propostos meios de resolver potenciais conflitos com trajetórias de outros robôs buscando sempre o meio mais rápido de realizar a respectiva tarefa.

Através da realização de experimentos com o método proposto foi possível validar as técnicas empregadas. O método proposto é capaz de resolver o problema de exploração de ambientes estruturados, realizando a coordenação dos múltiplos robôs respeitando requisitos de segurança, como a não colisão com outros robôs e obstáculos, e também

sendo capaz de finalizar a exploração para todas as situações propostas, sem levar o sistema a situações de bloqueio.

O método também demonstrou ser capaz de solucionar o problema de exploração para ambientes de dimensão relativamente elevada com até oito robôs, mesmo realizando inúmeras vezes o recálculo do planejamento de trajetórias ao longo de uma exploração.

As heurísticas utilizadas para a alocação de tarefas se destacaram quando confrontadas com uma alocação realizada de forma aleatória. Utilizando as heurísticas propostas, isoladamente ou de forma combinada, ficaram claras as reduções no tempo total de exploração.

O trabalho aqui apresentado também resultou na publicação de um artigo no XI Simpósio Brasileiro de Automação Inteligente [Ribeiro, Farines e Cury 2013]. O artigo tratou parcialmente dos métodos apresentados neste trabalho e mostrou alguns resultados preliminares da aplicação do método de exploração, os quais foram mais profundamente estudados nessa dissertação.

## 6.1 PERSPECTIVAS

Como trabalhos futuros, a partir do que foi apresentado nessa dissertação, propõe-se uma série de extensões e melhorias no método de exploração.

**Método para estabelecer prioridade** Ao realizar o planejamento de trajetória, o método proposto estabelece aleatoriamente a ordem de prioridades entre os robôs. Propõe-se, então, elaborar um método que estabeleça a ordem de prioridades baseando-se e características do sistema, como a impossibilidade de planejar a trajetória de determinado robô antes dos demais, focando a busca e diminuindo o número de vezes que essa ordem é proposta e avaliada;

**Ampliação do número de *features*** O método proposto é capaz de explorar ambientes estruturados formados por cinco tipos de *features*. Uma proposta de extensão do método é a inserção de novas *features*, como por exemplo a *feature* "sala", adicionando a ela conceitos como capacidade e exploração realizada por mais de um robô;

**Inserção de obstáculos dinâmicos** O método proposto nesse trabalho considera apenas ambientes estáticos. Propõe-se a inserção

de obstáculos dinâmicos, tais como portas, e a adaptação do método para que seja capaz de adotar uma estratégia utilizando conceitos de probabilidade, por exemplo, com intuito de buscar os conjuntos de caminhos mais rápidos para execução as tarefas;

**Mapas locais** Uma das considerações do método proposto é que os múltiplos robôs iniciam a exploração com referências globais da posição dos demais robôs, podendo assim incorporar os dados coletados do ambiente diretamente a um único mapa global. Uma proposta de trabalho futuro seria a eliminação dessa consideração, fazendo com que os múltiplos robôs iniciem a exploração em locais desconhecidos pelos demais. Dessa forma, mapas locais seriam construídos por cada um dos robôs e um método para fusão desses mapas desenvolvido.

Além dos itens propostos relacionados ao método de exploração, outra perspectiva futura para o trabalho é a sua implementação em uma planta real com protótipos de robôs móveis. Durante o período de desenvolvimento desse trabalho, foram também construídos pelo grupo de pesquisa o qual esse trabalho está inserido, pequenos robôs móveis equipados com *encoders*, sensores de proximidade e acelerômetro e uma bancada de testes com obstáculos móveis.

# REFERÊNCIAS

BAILEY, T.; NEBOT, E. Localisation in large-scale environments. *Robotics and Autonomous Systems*, Elsevier, v. 37, n. 4, p. 261–281, 2001.

BANDO, S.; YUTA, S. Use of the parallel and perpendicular characteristics of building shape for indoor map making and positioning. In: IEEE. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. [S.l.], 2010. p. 4318–4323.

BEEVERS, K.; HUANG, W. Loop closing in topological maps. In: IEEE. *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. [S.l.], 2005. p. 4367–4372.

BENNEWITZ, M. Mobile robot navigation in dynamic environments. *Unpublished doctoral dissertation, Alberg Ludwig University, Dept of Computer Science, Freiburg*, Citeseer, 2004.

BERG, M. D. et al. *Computational geometry: algorithms and applications*. [S.l.]: Springer, 2008. 191–218 p.

BURGARD, W. et al. Coordinated multi-robot exploration. *Robotics, IEEE Transactions on*, IEEE, v. 21, n. 3, p. 376–386, 2005.

CARVALHO, J. Modelagem e síntese para coordenação de sistemas multi-robôs baseada numa estrutura de jogo. *Dissertação de mestrado, Universidade Federal de Santa Catarina, Departamento de Automação e Sistemas, Florianópolis/SC*, 2012.

CHOSET, H. et al. *Principles of robot motion: theory, algorithms, and implementations*. [S.l.]: MIT press, 2005.

FILLIAT, D.; MEYER, J.-A. Map-based navigation in mobile robots:: I. a review of localization strategies. *Cognitive Systems Research*, Elsevier, v. 4, n. 4, p. 243–282, 2003.

GERKEY, B.; VAUGHAN, R. T.; HOWARD, A. The player/stage project: Tools for multi-robot and distributed sensor systems. In: *Proceedings of the 11th international conference on advanced robotics*. [S.l.: s.n.], 2003. v. 1, p. 317–323.

GERKEY, B. P.; MATARIĆ, M. J. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, SAGE Publications, v. 23, n. 9, p. 939–954, 2004.

GONG, F.; WANG, X. Robot path-planning based on triangulation tracing. In: IEEE. *Intelligent Information Technology Application Workshops, 2008. IITAW'08. International Symposium on*. [S.l.], 2008. p. 713–716.

GONZÁLEZ-BANOS, H. H.; HSU, D.; LATOMBE, J.-C. Motion planning: Recent developments. *Autonomous Mobile Robots: Sensing, Control, Decision-Making and Applications*, Boca Raton, FL: CRC Press, 2006.

HOUGEN, D. et al. A miniature robotic system for reconnaissance and surveillance. In: IEEE. *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*. [S.l.], 2000. v. 1, p. 501–507.

JI, X. et al. A decision-theoretic active loop closing approach to autonomous robot exploration and mapping. *RoboCup 2008: Robot Soccer World Cup XII*, Springer, p. 507–518, 2009.

KAPLOW, R.; ATRASH, A.; PINEAU, J. Variable resolution decomposition for robotic navigation under a pomdp framework. In: IEEE. *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. [S.l.], 2010. p. 369–376.

LATOMBE, J. *Robot motion planning*. [S.l.]: Springer, 1990.

LEE, T.; BAEK, S.; OH, S. Sector-based maximal online coverage of unknown environments for cleaning robots with limited sensing. *Robotics and Autonomous Systems*, Elsevier, v. 59, n. 10, p. 698–710, 2011.

MARJOVI, A.; MARQUES, L. Multi-robot olfactory search in structured environments. *Robotics and Autonomous Systems*, Elsevier, v. 59, n. 11, p. 867–881, 2011.

MEYER, J.-A.; FILLIAT, D. Map-based navigation in mobile robots:: Ii. a review of map-learning and path-planning strategies. *Cognitive Systems Research*, Elsevier, v. 4, n. 4, p. 283–317, 2003.

NAGATANI, K. et al. Multirobot exploration for search and rescue missions: A report on map building in robocuprescue 2009. *Journal of Field Robotics*, Wiley Online Library, v. 28, n. 3, p. 373–387, 2011.

NILSSON, N. Principles of artificial intelligence. *Symbolic Computation, Berlin: Springer, 1982*, v. 1, 1982.

OSTERGARD, E. *The Java Player Client*. 2001.

PAVEI, J. Coordenação em sistemas multi-robôs utilizando métodos baseados em autômatos. *Dissertação de mestrado, Universidade Federal de Santa Catarina, Departamento de Automação e Sistemas, Florianópolis/SC*, 2011.

QUOTTRUP, M.; BAK, T.; ZAMANABADI, R. Multi-robot planning: A timed automata approach. In: IEEE. *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. [S.l.], 2004. v. 5, p. 4417–4422.

RIBEIRO, V.; FARINES, J.; CURY, J. Exploração de ambientes desconhecidos estruturados com sistemas multi-robôs coordenados. In: SBA. *XI Simpósio Brasileiro de Automação Inteligente - SBAI. XI Conferência Brasileira de Dinâmica, Controle e Aplicações - DINCON. Outubro, 2013. Fortaleza-CE*. [S.l.], 2013.

SAVELLI, F.; KUIPERS, B. Loop-closing and planarity in topological map-building. In: IEEE. *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. [S.l.], 2004. v. 2, p. 1511–1517.

SICILIANO, B.; KHATIB, O. *Springer handbook of robotics*. [S.l.]: Springer, 2008.

SLEUMER, N. H.; TSCHICHOLD-GüRMAN, N. *Exact Cell Decomposition of Arrangements used for Path Planning in Robotics*. 1999.

STACHNISS, C. *Robotic mapping and exploration*. [S.l.]: Springer, 2009.

STACHNISS, C.; MOZOS, O.; BURGARD, W. Speeding-up multi-robot exploration by considering semantic place information. In: IEEE. *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. [S.l.], 2006. p. 1692–1697.

THRUN, S. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, Elsevier, v. 99, n. 1, p. 21–71, 1998.

THRUN, S. et al. A system for volumetric robotic mapping of abandoned mines. In: IEEE. *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on.* [S.l.], 2003. v. 3, p. 4270–4275.

THRUN, S. et al. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, Morgan Kaufmann, p. 1–35, 2002.

TOMATIS, N.; NOURBAKHSH, I.; SIEGWART, R. Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous systems*, Elsevier, v. 44, n. 1, p. 3–14, 2003.

VALLEJO, D. et al. A multi-agent architecture for multi-robot surveillance. *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*, Springer, p. 266–278, 2009.

WURM, K.; STACHNISS, C.; BURGARD, W. Coordinated multi-robot exploration using a segmentation of the environment. In: IEEE. *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on.* [S.l.], 2008. p. 1160–1165.

YAMAUCHI, B. A frontier-based approach for autonomous exploration. In: IEEE. *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on.* [S.l.], 1997. p. 146–151.

YAMAUCHI, B. Frontier-based exploration using multiple robots. In: ACM. *Proceedings of the second international conference on Autonomous agents.* [S.l.], 1998. p. 47–53.